普通高等教育“十一五”国家级规划教材·配套参考书
高等学校工程创新型“十二五”规划计算机教材

计算机组成原理学习指导与课程设计

罗克露　纪禄平　吉家成　编

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书是普通高等教育“十一五”国家级规划教材《计算机组成原理》（第 2 版）（ISNB 978-7-121-09546-7）的配套辅导书。全书分为 4 章，内容包括：计算机组成原理的实验和课程设计介绍；创新活动，涉及参加电子设计大赛等的建议和简单的指导；主教材各章的重点、难点解析，以及各章的习题解答及解题思路；考研辅导，包括考研内容复习和考研真题题解。

本书可以作为高等学校学习“计算机组成原理”课程的辅导教材，也可供相关专业学生考研参考。

图书在版编目(CIP)数据

计算机组成原理学习指导与课程设计/罗克露，纪禄平，吉家成编. —北京：电子工业出版社，2011.6
高等学校工程创新型“十二五”规划计算机教材
ISBN 978-7-121-13748-8

I. ①计… II. ①罗… ②纪… ③吉… III. ①计算机组成原理－高等学校－教学参考资料 IV. ①TP301

中国版本图书馆 CIP 数据核字(2011)第 103353 号

策划编辑：章海涛
责任编辑：章海涛
印　　刷：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：11　字数：269 千字
印　　次：2011 年 6 月第 1 次印刷
印　　数：3 000 册　　定价：25.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010)88258888。

前　言

本书是与普通高等教育“十一五”国家级规划教材《计算机组成原理》（第 2 版）（电子工业出版社 2010 年 2 月出版，ISNB 978-7-121-09546-7）配套的辅助教材。

本书内容丰富，主要有 4 章内容。

第 1 章，计算机组成原理的实验和课程设计介绍。

第 2 章，创新活动，涉及参加电子设计大赛等的建议和简单的指导。

第 3 章，主教材各章的习题解答及解题思路。

第 4 章，考研辅导，包括考研内容复习和考研真题题解。

期望本书能够为增强读者的实践能力、创新能力、分析问题和解决问题的能力提供帮助。

本书第 1 章由吉家成编写，第 2 章由纪禄平编写，第 3 章、第 4 章由罗克露编写，全书由罗克露统稿。

鉴于作者水平有限，经验不足，书中一定存在不少错误与疏漏之处，恳请同行和读者批评指正。

作　者

于成都・电子科技大学

目　　录

第 1 章　实验及课程设计

“计算机组成原理”是计算机专业的基础课和必修课，通常包括两个教学环节：理论教学和实验教学。其中，实验教学是本课程的一个重要组成部分，只有通过亲自动手做实验，才能熟悉和掌握计算机的基本工作原理，加深对书本上知识的理解和认识，同时为后续课程的学习打下良好的基础。

每个学校的教学计划、实验条件和学时数各不相同，选用的实验平台也各异，主要的平台有两类，一类是“基于 EDA 技术的组成原理实验箱”，另一类是“专用的组成原理实验箱”。目前，国内很多高校选用基于 EDA 技术的实验平台。EDA（Electronic Design Automation）技术是近年来发展迅猛的一种新的设计硬件电路的方法，它利用硬件描述语言 Verilog HDL 或 VHDL 对电路进行开发设计，其开发流程是：先在计算机上进行建模和仿真，然后通过相应的接口下载到实验箱上的 FPGA/CPLD 芯片上进行验证。这种实验方法主要以算法方式对硬件电路进行描述，要求设计者具有基本的硬件知识，由于硬件描述语言 Verilog HDL 与普通的C 语言在语法上有很多相似之处，因此这对于掌握了一定编程方法的学生来说非常容易上手。而且，硬件描述语言也是IC（集成电路）设计的重要工具，通过学习和使用，使学生在硬件、软件设计上得到了锻炼，对学生今后的就业是大有益处的。

1.1　实验平台及说明

1.1.1　嵌入式硬件平台

（1）核心适配板

实验用的可编程逻辑芯片是美国 XILINX 公司的 SPARTAN XC2S200，属于 FPGA 型可编程逻辑器件，内含 20 万门电路，具有可编程接口（JTAG），通过并口与计算机相连，该芯片的内容可以反复擦写。

（2）输入/输出接口

① 按键开关：按下为低电平，弹起为高电平，用于输入单个脉冲信号。

② 拨码开关：开关向上为高电平，向下为低电平，用于输入高低电平信号。

③ 发光二极管：高电平点亮，低电平熄灭，用于观察输出信号的高低电平。

④ 40 MHz 的时钟基准信号，用于各种信号的分频时钟。

1.1.2　软件集成开发环境

本实验系统的开发软件采用美国 XILINX 公司的 ISE 集成开发环境。下面以一个实例来介绍其软件的开发流程。

1．创建工程

双击桌面上的“Xilinx ISE 7.1”图标；选择“File”中的“New Project”，显示如图 1.1

所示的对话框，从中填写“工程项目名”和文件存放的路径；单击“下一步”按钮，显示如图1.2 所示的对话框，正确选择所用芯片的类型、封装形式、I/O 引脚的数目等信息，再选择综合工具（Synthesis Tool）。

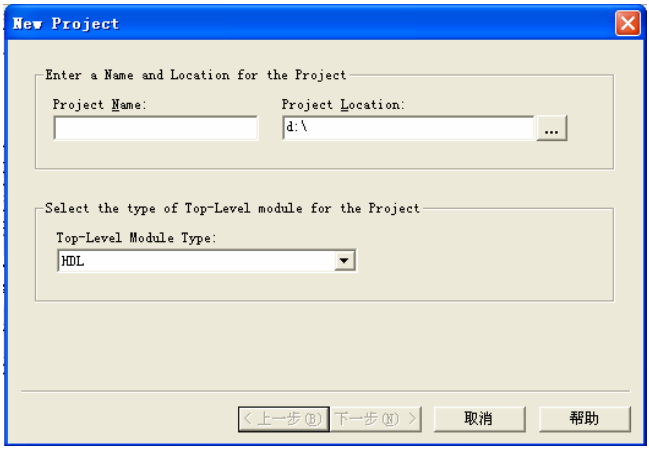


图 1.1　建立项目工程

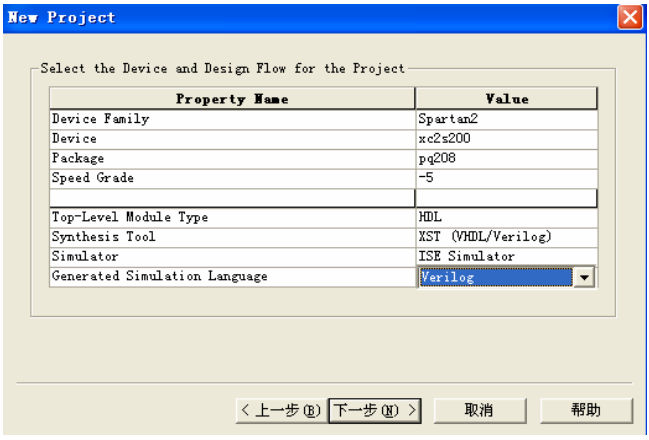


图 1.2　配置芯片信息

2．设计文件的输入

在如图1.3所示的对话框中输入文件名，同时选择左框中的“Verilog Module”；然后输入所编写的源程序代码。

3．约束（“引脚绑定”），定位I/O引脚的具体编号

（1）在“Process View”框中单击“User Constraints”前的“+”，双击“Assign Package Pins”。

（2）在“Design Browser”框中，选择“I/O Pins”。

（3）在“Design Object List…”中的“Loc”栏内填入芯片的引脚编号。注意，在引脚编号前加上字母 p。

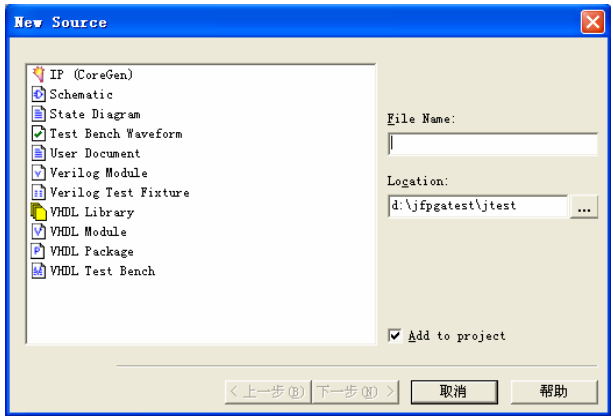


图 1.3　文件配置

4．综合（Synthesize），检查所输入的源程序是否有语法错误

在“Process View”框中选择“Synthesize-XST”。

5．实现（Implement），检查引脚约束是否有错误

在“Process View”框中选择“Implement Design”。

6．将程序下载至芯片

（1）在“Process View”框中选择“Configure Device（Impact）”。

（2）选择“Boundary-Scan Mode”。

（3）再选择“Automatically connect to cable….”（注意此时必须将实验目标板用电缆与计算机并口相连，同时打开实验设备的电源），显示如图1.4所示。

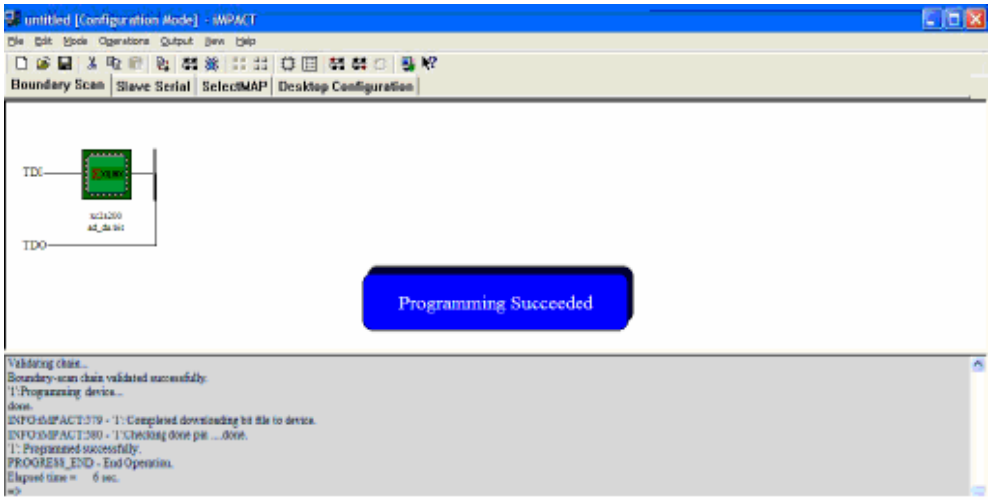


图 1.4　程序下载

（4）双击“XC2S200 芯片”图标，在弹出的对话框中选取需要下载的 Bit 文件。

（5）在“XC2S200 芯片”图标上单击右键，选择“Programme”，即可将程序下载到 FPGA 芯片中。

1.2 实验

1.2.1 运算器ALU设计实验

1．实验目的

（1）熟悉运算器（ALU）的组成结构及工作原理。

（2）掌握运算器模块的设计方法。

（3）掌握多个 ALU 级联和进位的连接方法。

（4）熟悉数据的暂存和分时传送的控制方法。

2．实验内容

（1）设计一个 4 位运算器，要求该运算器具有算术运算（加、减）和逻辑运算（与、或、异或、清零）功能。

（2）调用上面已调试好的 4 位 ALU 模块，把它扩展成 8 位 ALU，实现 8 位算术运算和逻辑运算功能。

（3）用一个数据通道（8 位数据宽度）和相应的控制信号，实现两个 8 位数据的分时输入。

（4）编写和调试相应的程序模块并下载到实验箱进行测试。

运算器实验整体框图如图1.5所示。

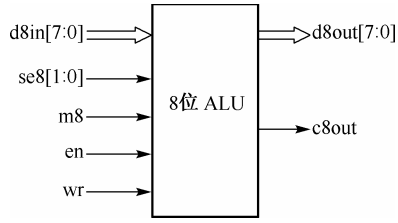


图 1.5　运算器实验框图

3．实验原理

运算器实验原理图如图1.6 所示。

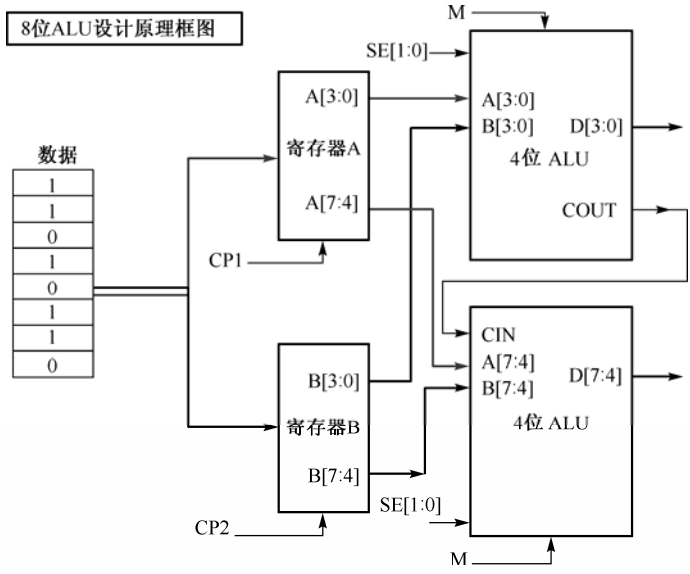


图 1.6　运算器实验原理图

4．实验核心代码

```
module alu8(d8in,m8,se8,en,wr,c8out,d8out);
parameter D_WIDTH =8;

    input   [D_WIDTH-1:0] d8in;          //数据输入
    input   m8;                          //运算模式输入：算术/逻辑
    input   en;                          //数据输入控制电平
    input   wr;                          //数据输入控制脉冲
    input   [1:0] se8;                   //运算方式输入：加、减/与、或、异或、清零
    output   c8out;                      //进位输出
    output   [D_WIDTH-1:0] d8out;        //运算数据输出
    reg   [D_WIDTH-1:0] ra;              //数据暂存器
    reg   [D_WIDTH-1:0] rb;              //数据暂存器
    wire   rcout;

    //下面程序说明：由于数据输入通道只有 8 位，因此只有采用分时的方式输入数据。
    //即：第一个节拍，当 en=0 时，由控制脉冲 wr 将外部输入数据打入暂存器 ra;
    //第二个节拍，当 en=1 时，由控制脉冲 wr 将外部输入数据打入暂存器 rb;
    always @(posedge wr)
    if (!en) ra<=d8in;
        else   rb<=d8in;
    //下面程序说明：通过两条实例化语句，调用 4 位运算器模块，实现一个 8 位的运算器
    alu4 alu8l(.a(ra[3:0]),.b(rb[3:0]),.m(m8),.cin(0),.se(se8),.cout(rcout),.d(d8out[3:0]));
    alu4 alu8h(.a(ra[7:4]),.b(rb[7:4]),.m(m8),.cin(rcout),.se(se8),.cout(c8out),.d(d8out[7:4]));
```

下面是 4 位运算器模块的源程序：

```
module alu4(a,b,m,cin,se,cout,d);
parameter DATA_WIDTH =4;

    input   [DATA_WIDTH-1:0] a;
    input   [DATA_WIDTH-1:0] b;
    input   [1:0] se;
    input   cin;
    input   m;
    output   cout;
    output   [DATA_WIDTH-1:0] d;
    reg   cout;
    reg   [DATA_WIDTH-1:0] d;

    always @(a or b or cin or m or se)
    begin
        if(!m)
        begin
            case(se)
                    2'b00 :   {cout,d}=a+b+cin;
```

```
                2'b01 :   {cout,d}=a-b-cin;
                default:   d=0;
                endcase
        end
        else
        begin
            case(se)
                2'b00:        d=a&b;
                2'b01:        d=a|b;
                2'b10:        d=a^b;
                2'b11:        d=0;
            endcase
        end
    end
endmodule
```

1.2.2 存储器设计实验

1．实验目的

（1）掌握存储器模块的设计方法。

（2）掌握存储器容量的字扩展和位扩展技术。

（3）掌握存储器片选信号的控制方法。

2．实验内容

设计一个 32×8 的静态存储器 SRAM，能够对存储单元进行随机读写。其中 32 表示地址线寻址存储单元空间的大小，8 表示存储单元的数据位数。

具体要求如下：

（1）设计一个 16×4 位的 SRAM 程序模块。

（2）利用 16×4 位的 SRAM 程序模块，通过“实例化”的程序设计方法，再把它扩展成一个 32×8 位的 SRAM。

（3）随机写入和读出 SRAM 中的数据。

（4）编写和调试相应的程序模块，并下载到实验箱进行测试。

3．实验原理

做这个实验关键是要掌握“存储器容量的字扩展和位扩展”的工作原理，搞清楚计算机中三个总线的数据流向以及数据总线上“三态”的基本概念和控制方法，然后画出整个实验的系统框图，通过“实例化”的编程调用，实现其功能。图1.7 为存储器实验原理图的参考图。

说明：数据总线是双向的，为了简化实验的操作过程，我们把它定义成单向的，即分成数据输入总线和数据输出总线。

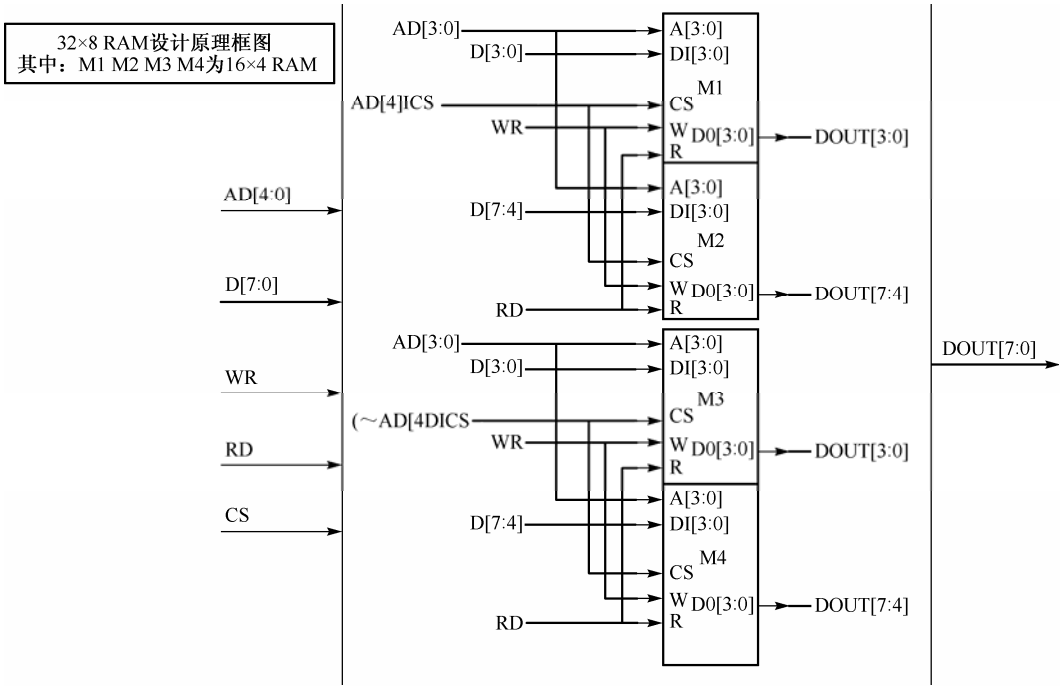


图 1.7　存储器实验原理图

4．实验核心代码

```
module ram32_8(din8,add32,wr8,rd8,cs8,dout8);
parameter D8_WIDTH = 8;                          //定义数据总线的宽度 8 位
parameter A32_WIDTH =5;                          //定义地址总线的宽度 5 位

    input   [D8_WIDTH-1:0]   din8;               //数据输入总线 8 位
    input   [A32_WIDTH-1:0] add32;               //地址输入总线 5 位
    input   wr8;                                 //写控制输入信号线
    input   rd8;                                 //读控制输入信号线
    input   cs8;                                 //片选控制输入信号线
    output   [D8_WIDTH-1:0]     dout8;           //数据输出总线 8 位
    wire     csl;                          // csl 为  0～15 单元的片选信号(由模块 M1 和 M2 组成)
    wire     csh;                          // csh 为 16～31 单元的片选信号(由模块 M3 和 M4 组成)

    assign csl=(!cs8)?add32[4]:1'b1;
    assign csh=(!cs8)?(~add32[4]):1'b1;

    ram164 raml30(.din(din8[3:0]),.addr(add32[3:0]),.wr(wr8),.rd(rd8),.cs(csl),.dout(dout8[3:0])); //模块 M1;
    ram164 raml74(.din(din8[7:4]),.addr(add32[3:0]),.wr(wr8),.rd(rd8),.cs(csl),.dout(dout8[7:4])); //模块 M2;
    ram164 ramh30(.din(din8[3:0]),.addr(add32[3:0]),.wr(wr8),.rd(rd8),.cs(csh),.dout(dout8[3:0])); //模块 M3;
    ram164 ramh74(.din(din8[7:4]),.addr(add32[3:0]),.wr(wr8),.rd(rd8),.cs(csh),.dout(dout8[7:4])); //模块 M4;
Endmodule
```

下面是 16×4 存储器模块的源程序。

```
    // din     数据输入总线 4 位
    // addr    地址输入总线 4 位
    // wr      写控制输入信号线
    // rd      读控制输入信号线
    // cs      片选控制输入信号线
    // dout    数据输出总线 4 位
module ram164(din,addr,wr,rd,cs,dout);
    parameter D_WIDTH = 4;
    parameter A_WIDTH = 4;

    input   [D_WIDTH-1:0]   din;
    input   [A_WIDTH-1:0]   addr;
    input   wr,rd,cs;
    output  [D_WIDTH-1:0]   dout;
    reg   [D_WIDTH-1:0] ram [(2**A_WIDTH)-1:0];
    wire   [D_WIDTH-1:0] dout;

    always @(posedge wr)                          //存储器写控制
    if (!cs)
        ram[addr] <= din;

    assign dout = (!(rd||cs))?ram[addr]:4'bzzzz;       //存储器读控制
endmodule
```

1.3 课程设计及指导

课程设计题目：模型机指令系统的设计。

本课程设计的目的是，让学生掌握设计一个体现计算机基本功能的简单指令系统的方法。

计算机程序由一系列机器指令组成，一台计算机所能执行的所有指令的集合称为该机的指令系统。通常，一条指令由“操作码（OP）”和“地址码（AD）”两部分组成。其中，“操作码”表示该指令进行什么类型的操作，如加法、减法、逻辑与、逻辑或等；“地址码”反映操作数的有关信息，指明操作数的来源、运算结果存放何处等。

在指令格式设计时应该考虑以下问题：

（1）指令的字长需要多少位

指令的字长位数越多，所能表示的操作信息和地址信息就越多，指令的功能就越丰富。但是位数多则指令所占的存储空间加大，读取指令所需要的时间加长，所以在设计指令格式时，需要综合考虑。由于在计算机中存储单元是按字节编址的，因此指令的字长位数通常是 8 的倍数。

（2）操作码需要多少位

操作码的位数将决定操作类型的多少，显然位数越多，所能表示的操作种类就越多。当指令的字长位数确定以后，将根据指令操作类型多少和地址码的位数多少，加以权衡考虑。

指令中的操作数可能在 CPU 的寄存器中，也可能在内存中，因此在指令中需要给出相应的地址信息，可能是 CPU 中的寄存器号，也可能是存储单元的地址码。由地址码得到操作数，这就涉及地址的寻址方式。所谓寻址方式，就是如何对地址字段进行解析，以便找到操作数。一个指令系统具有哪几种寻址方式，地址是以什么方式给出的，这些都需要在设计指令系统时认真研究。

下面以一个模型机为实例，详细分析一个定长为 16 位的指令系统的设计过程。

1．指令格式

指令码用 IR（16 位）表示，模型机指令格式见图1.8。

图 1.8　模型机指令格式

指令码IR的16位二进制数分成5个字段，定义如下：

① IR15，IR14，IR13，IR12——用于操作码的类型控制。

② IR11，IR10，IR9——用于源操作数寻址方式的控制。

③ IR8，IR7，IR6——用于源寄存器号的控制。

④ IR5，IR4，IR3——用于目的操作数寻址方式的控制。

⑤ IR2，IR1，IR0——用于目的寄存器号的控制。

2．操作类型

操作码类型控制编码为 4 位，可以表示 16 种指令，见表 1-1。

表 1-1

操作码类型字段编码 IR15　IR14　IR13　IR12	操作码类型说明（汇编符号）
0　0　0　0	传送指令　MOV
0　0　0　1	加法指令　ADD
0　0　1　0	减法指令　SUB
0　0　1　1	逻辑与指令　AND
0　1　0　0	逻辑或指令　OR
0　1　0　1	逻辑异或指令　EOR
0　1　1　0	取反指令　COM
0　1　1　1	求补指令　NEG
1　0　0　0	加 1 指令　INC
1　0　0　1	减 1 指令　DEC
1　0　1　0	左移指令　SL
1　0　1　1	右移指令　SR
1　1　0　0	转移指令　JMP
1　1　0　1	返回指令　RST
1　1　1　0	转子指令　JSR
1　1　1　1	保留

3．寻址方式

（1）寄存器号

寄存器号编码为 3 位，可以表示 8 种寄存器，见表 1-2。

表 1-2

寄存器号字段编码 IR8　IR7　IR6（源） IR2　IR1　IR0（目的）	功能说明（汇编符号）
0　0　0	通用寄存器　　R0
0　0　1	通用寄存器　　R1
0　1　0	通用寄存器　　R2
0　1　1	通用寄存器　　R3
1　0　0	堆栈指针　　SP
1　0　1	程序状态字　　PSW
1　1　0	程序计数器　　PC
1　1　1	保留

（2）操作数寻址方式

该字段编码为 3 位，可以表示 8 种寻址方式，见表 1-3。

表 1-3

寻址方式字段编码 IR11　IR10　IR9（源） IR5　IR4　IR3（目的）	功能说明（汇编符号）
0　0　0	寄存器寻址　　R
0　0　1	寄存器间接寻址　　(R)
0　1　0	自减型寄存器间接寻址　　−(R)
0　1　1	自增型寄存器间接寻址　　(R)+
1　0　0	寄存器、存储器双重间接寻址　　@(R)+
1　0　1	变址寻址　　X(R)
1　1　0	跳步寻址　　SKP
1　1　1	保留　　RESERVE

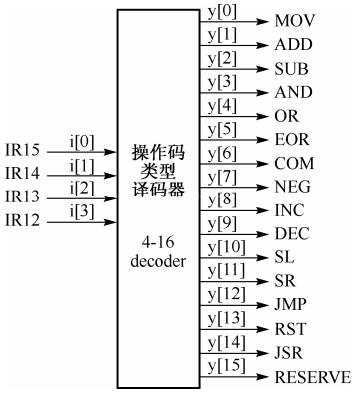


图 1.9　操作码类型译码器

4．设计与实现

从前面指令格式的定义和分析可知，该指令码由 5 个字段组成，实际上就对应 5 个译码器，通过对它们的译码，产生相应的输出控制信号。注意：所有输出信号低电平有效。

（1）操作码类型译码器（见图 1.9）

输入（4 位）： IR15，IR14，IR13，IR12

输出（16 位）：MOV，ADD，SUB，AND，OR，EOR，COM，NEG，INC，DEC，SL，SR，

JMP，RST，JSR，RESERVE

```
// “操作码类型译码器”核心代码
    reg[3:0]   i;                                   //IR15,IR14,IR13,IR12
    reg[15:0] y;                                    //RESERVE,JSR,RST,JMP,SR,SL,DEC,INC,NEG,
                                                    //COM,EOR,OR,AND,SUB,ADD,MOV
    case(i)
        16'd0:     y=16'b1111111111111110;          //MOV 指令有效
        16'd1:     y=16'b1111111111111101;          //ADD 指令有效
        16'd2:     y=16'b1111111111111011;          //SUB 指令有效
        16'd3:     y=16'b1111111111110111;          //AND 指令有效
        16'd4:     y=16'b1111111111101111;          //OR 指令有效
        16'd5:     y=16'b1111111111011111;          //EOR 指令有效
        16'd6:     y=16'b1111111110111111;          //COM 指令有效
        16'd7:     y=16'b1111111101111111;          //NEG 指令有效
        16'd8:     y=16'b1111111011111111;          //INC 指令有效
        16'd9:     y=16'b1111110111111111;          //DEC 指令有效
        16'd10:    y=16'b1111101111111111;          //SL 指令有效
        16'd11:    y=16'b1111011111111111;          //SR 指令有效
        16'd12:    y=16'b1110111111111111;          //JMP 指令有效
        16'd13:    y=16'b1101111111111111;          //RST 指令有效
        16'd14:    y=16'b1011111111111111;          //JSR 指令有效
        default:   y=16'b1111111111111111;          //RESERVE 保留指令
    endcase
```

（2）源操作数寻址方式译码器（见图1.10）

输入（3 位）：IR11，IR10，IR9

输出（8 位）：S_AD_REG，S_AD_INDI，S_AD_DECR，S_AD_INCR，
S_DOUB_INDI，S_AD_VARI，S_AD_SKP，INVALID

```
// “源操作数寻址方式译码器”核心代码
    reg[2:0] i;                             //IR11,IR10,IR9
    reg[7:0] y;                             //INVALID,S_AD_SKP,S_AD_VARI,S_DOUB_INDI,
                                            //S_AD_INCR,S_AD_DECR,S_AD_INDI,S_AD_REG
    case(i)
        8'd0:     y=8'b11111110;            //源操作数选择寄存器寻址方式
        8'd1:     y=8'b11111101;            //源操作数选择寄存器间接寻址方式
        8'd2:     y=8'b11111011;            //源操作数选择自减型寄存器间接寻址方式
        8'd3:     y=8'b11110111;            //源操作数选择自增型寄存器间接寻址方式
        8'd4:     y=8'b11101111;            //源操作数选择寄存器、存储器双重间接寻址方式
        8'd5:     y=8'b11011111;            //源操作数选择变址寻址方式
        8'd6:     y=8'b10111111;            //源操作数选择跳步寻址方式
        default:  y=8'b11111111;            //无效
    endcase
```

（3）源寄存器号译码器（见图1.11）

输入（3 位）：IR8，IR7，IR6

输出（8 位）：S_R0，S_R1，S_R2，S_R3，S_SP，S_PSW，S_PC，INVALID

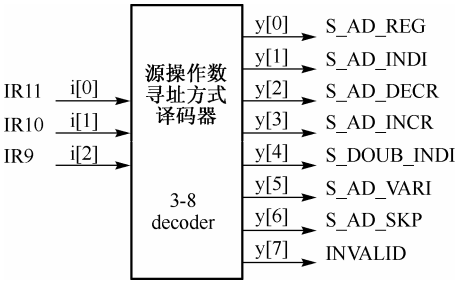


图 1.10　源操作数寻址方式译码器

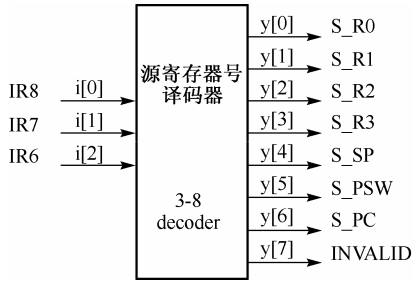


图 1.11　源寄存器号译码器

```
//“源寄存器号译码器”核心代码
    reg[2:0] i;                     //IR8,IR7,IR6
    reg[7:0] y;                     //INVALID,S__PC,S_PSW,S_SP,S_R3,S_R2,S_R1,S_R0

    case(i)
        8'd0:    y=8'b11111110;     //源寄存器选中 R0 有效
        8'd1:    y=8'b11111101;     //源寄存器选中 R1 有效
        8'd2:    y=8'b11111011;     //源寄存器选中 R2 有效
        8'd3:    y=8'b11110111;     //源寄存器选中 R3 有效
        8'd4:    y=8'b11101111;     //源堆栈指针选中 SP 有效
        8'd5:    y=8'b11011111;     //源程序状态字选中 PSW 有效
        8'd6:    y=8'b10111111;     //源程序计数器选中 PC 有效
        default: y=8'b11111111;     //无效
    endcase
```

（4）目的操作数寻址方式译码器（见图1.12）

输入（3 位）：IR5，IR4，IR3

输出（8 位）：D_AD_REG，
D_AD_INDI，D_AD_DECR，
D_AD_INCR，D_DOUB_INDI，
D_AD_VARI，D_AD_SKP，INVALID

```
//“目的操作数寻址方式译码器”核心代码
    reg[2:0] i;                     //IR5,IR4,IR3
    reg[7:0] y;                     //INVALID,D_AD_SKP,D_AD_VARI,D_DOUB_INDI,
                                    //D_AD_INCR,D_AD_DECR,D_AD_INDI,D_AD_REG
    case(i)
        8'd0:    y=8'b11111110;     //目的操作数选择寄存器寻址方式
        8'd1:    y=8'b11111101;     //目的操作数选择寄存器间接寻址方式
        8'd2:    y=8'b11111011;     //目的操作数选择自减型寄存器间接寻址方式
        8'd3:    y=8'b11110111;     //目的操作数选择自增型寄存器间接寻址方式
        8'd4:    y=8'b11101111;     //目的操作数选择寄存器、存储器双重间接寻址方式
        8'd5:    y=8'b11011111;     //目的操作数选择变址寻址方式
        8'd6:    y=8'b10111111;     //目的操作数选择跳步寻址方式
        default: y=8'b11111111;     //无效
    endcase
```

（5）目的寄存器号译码器（见图1.13）

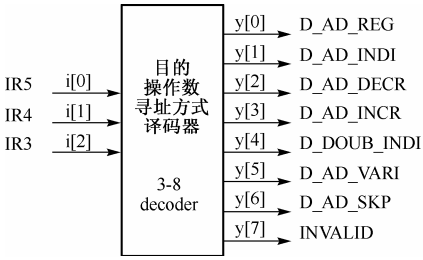


图 1.12　目的操作数寻址方式译码器

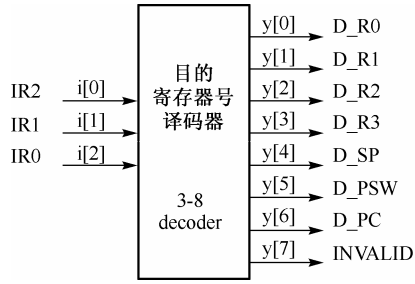


图 1.13　目的寄存器号译码器

输入（3 位）：IR2，IR1，IR0

输出（8 位）：D_R0，D_R1，D_R2，D_R3，D_SP，D_PSW，D_PC，INVALID

```
//“目的寄存器号译码器”核心代码
    reg[2:0] i;                         //IR2,IR1,IR0
    reg[7:0] y;                         //INVALID,D__PC,D_PSW,D_SP,D_R3,D_R2,D_R1,D_R0

    case(i)
        8'd0:    y=8'b11111110;         //目的寄存器选中 R0 有效
        8'd1:    y=8'b11111101;         //目的寄存器选中 R1 有效
        8'd2:    y=8'b11111011;         //目的寄存器选中 R2 有效
        8'd3:    y=8'b11110111;         //目的寄存器选中 R3 有效
        8'd4:    y=8'b11101111;         //目的堆栈指针选中 SP 有效
        8'd5:    y=8'b11011111;         //目的程序状态字选中 PSW 有效
        8'd6:    y=8'b10111111;         //目的程序计数器选中 PC 有效
        default: y=8'b11111111;         //无效
    endcase
```

在此基础上，学生可以针对具体操作类型和具体寻址方式的实现过程进行更详细的设计。

第 2 章　创 新 活 动

“计算机组成原理”作为一门大学本科计算机专业的核心硬件基础课程，在学习过程中不仅要牢固掌握以 CPU 为核心的计算机硬件系统基本理论，还要积极参加与课程相关的实践活动，锻炼自己分析和解决问题的能力，从而达到培养创新能力的目的。

现阶段，课程最典型的创新活动体现为参加各级各类电子设计竞赛。适合学生参加的电子竞赛比较多，大到全国性比赛、省市一级比赛，小到校级比赛，甚至还有企业举办的电子设计专题邀请赛等。这些竞赛看似千差万别，实质上，所有比赛对参赛者知识和创新设计能力的要求是一致的。下面将主要以“全国大学生电子设计竞赛”为对象，介绍电子竞赛的参赛事项，并结合具体题目做一些参赛指导，让读者对电子竞赛有一个基本了解，为选择性地参加这些竞赛做好准备，起到帮助学生积极参加创新活动、培养创新能力的作用。

2.1　全国大学生电子设计竞赛

全国大学生电子设计竞赛是教育部倡导的大学生学科竞赛之一，是面向大学生的群众性科技活动，目的在于推动高等学校促进信息与电子类学科课程体系和课程内容的改革，有助于高等学校实施素质教育，培养大学生的实践创新意识和基本能力、团队协作的人文精神和理论联系实际的学风；有助于学生工程实践素质的培养，提高学生针对实际问题进行电子设计制作的能力；有助于吸引、鼓励广大青年学生踊跃参加课外科技活动，为优秀人才的脱颖而出创造条件。

此项竞赛的特点是与高等学校相关专业的课程体系和课程内容改革密切结合的，以推动课程教学、教学改革和实验室建设工作。竞赛的特色是与理论联系实际的学风建设紧密结合，竞赛内容既有理论设计，又有实际制作，以全面检验和加强参赛学生的理论基础和实践创新能力。全国大学生电子设计竞赛的组织运行模式为：“政府主办、专家主导、学生主体、社会参与”，以充分调动各方面的参与积极性。

竞赛的内容和基本要求主要体现在下列三方面。

① 以电子电路（含模拟和数字电路）应用设计为主要内容，可以涉及模-数混合电路、单片机、可编程器件、EDA 软件工具和微机（主要用于开发）的应用。题目包括“理论设计”和“实际制作与调试”两部分。竞赛题目应具有实际意义和应用背景，并考虑到目前教学的基本内容和新技术的应用趋势，同时对教学内容和课程体系改革起一定的引导作用。

② 题目着重考核学生综合运用基础知识进行理论设计的能力，考核学生的创新精神和独立工作能力，考核学生的实验技能（制作、调试）。

③ 题目在难易程度方面，既要考虑使一般参赛学生能在规定的时间内完成基本要求，又能使优秀学生有发挥和创新的余地。

此项竞赛原则上安排在单数年的 9 月中旬举行，为期 4 天。竞赛以赛区为单位，统一组

织报名、竞赛、评审和评奖工作。此项竞赛鼓励设有信息与电子学科及相关专业或已开展电子设计科技活动的高等学校，积极组织学生参加。学生自愿组合，三人一队，由所在学校统一向赛区组委会报名，参赛队数由学校自行确定。为了鼓励不同类型的高校和不同专业或专业方向的学生都能参加竞赛，全国竞赛专家组根据命题原则，将统一编制若干竞赛题目，供参赛学生选用。竞赛所需场地、仪器设备、元器件或耗材原则上由参赛学校负责提供。

竞赛组委会还要求，参赛学生应是高等学校中具有正式学籍的全日制在校本科或专科学生。参赛学生必须按统一时间参加竞赛，按时开赛，准时交卷。各赛区组委会须按时收回学生的答卷（报告和制作实物）并及时封存，然后按规定交赛区专家组评审。在竞赛期间，参赛学生可以使用各种图书资料和计算机，但不得与队外人员讨论，教师必须回避，各赛区组委会还会组织巡视检查，以保证竞赛活动正常进行。在竞赛中，如发现辅导教师参与、队与队之间讨论、队员与队外人员讨论、不按规定时间发题和收卷及赛前泄题等违纪现象，将取消获奖名次，并通报批评。

全国大学生电子设计竞赛采取“一次竞赛、两级评奖”方式，评奖等级分为“赛区奖”和“全国奖”两种形式。

① 各赛区负责本赛区的评奖工作，赛区奖的评奖等级及各奖项获奖比例由各赛区根据实际情况自行确定。

② 赛区评审结束后，各赛区组委会将本赛区竞赛优秀参赛队的设计报告及有关评审材料报送全国组委会，报送的优秀参赛队数应严格控制在本赛区参赛队总数的 12%之内。全国组委会根据全国专家组的评审结果确定全国一、二等奖，获奖总数原则上不超过全国参赛队总数的 10%。

③ 全国大学生电子设计竞赛设立“赛区优秀组织奖”，对当年竞赛组织中表现出色的赛区组委会给予表彰奖励；同时设立“优秀征题奖”，对当年竞赛征题工作中表现突出的个人给予表彰奖励。

大学除了可以参加两年一度的普通电子设计竞赛以外，还可以参加各种专题邀请赛，如大学生电子设计竞赛组委会与英特尔公司联合组办的“嵌入式系统”专题邀请赛，以及教育部高教司、工信部人教司和大学生电子设计竞赛组委会联合举办的“信息安全技术”专题邀请赛。

2.1.1 嵌入式系统专题邀请赛

英特尔杯大学生电子设计竞赛嵌入式系统专题邀请赛，简称嵌入式系统专题邀请赛，由全国大学生电子设计竞赛组委会主办，上海市教委和上海交通大学承办，英特尔（中国）有限公司协办，并设置英特尔杯。

竞赛时间一般是每年的 3 月份到 6 月份，要求 3 月中下旬完成参赛队报名，7 月初完成参赛作品及相应的设计报告，并于 7 月中下旬组织专家评审评选出一、二、三等奖，同时在一等奖中评选出英特尔杯获奖队，随后“嵌入式系统专题邀请赛”组委会审定评审结果并公布最终获奖名单。

嵌入式系统专题邀请赛的参赛对象，为普通高校全日制在校本科学生，每个参赛队限三

人。每个参赛队可配备一名指导教师，指导教师的职责是：组织学生参加赛前培训，指导参赛队选题；确保竞赛期间嵌入式系统只能用于参赛作品的开发设计，不能挪作他用；负责竞赛过程中与学校及组委会之间的信息沟通。

此项竞赛统一采用大赛提供的基于英特尔凌动处理器的嵌入式系统，参赛队自主命题，自主设计，独立完成一个有一定功能的应用系统（竞赛作品）。竞赛形式采用开放式，不限定竞赛场所，参赛队利用课余时间，在规定时间内由参赛学生完成作品的设计、制作、调试及设计报告。

嵌入式系统专题邀请赛设置一、二、三等奖，原则上一等奖数目为参赛队数目的 8%，二、三等奖的数目分别为 16%和 36%。竞赛的评审结果由嵌入式系统专题邀请赛组委会审定，并在网上公布，获奖证书由嵌入式系统专题邀请赛组委会统一印制、颁发。

全国大学生电子设计竞赛“嵌入式系统专题邀请赛”组委会秘书处设在上海交通大学，联系人及联系方式如下。

官方网站：http://nuedc.sjtu.edu.cn。

联系方式：上海市闵行东川路 800 号上海交通大学电子信息学院楼群 1 号楼 409 室

联系人：蒋乐天　　邮政编码：200240　　电话/传真：021-34204354

电子邮件：nuedc@sjtu.edu.cn

2.1.2 信息安全技术专题邀请赛

本邀请赛是由教育部高教司、工业和信息化部人教司、由全国大学生电子设计竞赛组委会共同主办的全国性大学生科技竞赛活动，属于全国大学生电子设计竞赛的又一专题竞赛项目，其目的在于按照紧密结合教学实际，着重基础、注重前沿的原则，促进信息安全专业和课程的建设，引导高等学校在教学中注重培养大学生的创新能力、协作精神；加强学生动手能力的培养和工程实践的训练，提高学生综合素质，为优秀人才脱颖而出创造条件。

本邀请赛以邀约的方式组织比赛，邀请的参赛主体是具有研究生院、设有信息安全专业的院校，全国约 50 所左右。参赛学生应是高等学校中具有正式学籍的全日制在校本科生。选题类型分为硬件、软件两类，硬件类题目是在 TI 和 XILINX 公司提供的平台上完成信息安全相关作品，软件类题目定义为信息安全防护技术。参赛内容要求必须涉及网络安全和通信安全，但对参赛题目不做具体限定，特别强调理论、技术或应用的创新。

此项竞赛时间一般是每年 3 月中下旬开始，8 月中旬结束。竞赛形式采用开放式竞赛，每三人组成一个参赛队，同一学校最多可组 3 个队，其中 TI 公司平台 1 个队、XILINX 公司平台 1 个队、软件开发 1 个队。参赛学生应该独立完成竞赛作品，可以使用各种图书资料和网络资源。此外，为了更好地开展竞赛工作，在比赛前，TI 公司、XILINX 公司还会组织资深工程师为参赛学生及指导教师进行专项培训，确保各参赛队能熟练使用比赛专用平台。

一等奖获奖总数、二等奖获奖总数、三等奖获奖总数原则上分别不超过全国实际参赛队总数的 8%、16%和 36%。一般要求在 8 月上旬完成参赛作品及设计报告等资料的提交，8 月中下旬组委会组织信息安全技术专题邀请赛专家组评审各队提交的参赛作品，随后由竞赛组委会公布获奖名单。

2.2 毕昇杯全国电子创新设计竞赛

毕昇杯全国电子创新设计竞赛由北京精仪达盛科技有限公司主办，旨在纪念中华民族的“四大发明”的伟大创举，弘扬民族创新的精神，传承中国悠久的历史文化，发扬中华民族发明创新的传统美德，振兴中华民族。

此项电子设计竞赛的另一个目的是帮助中国高等院校全面提高大学生电子毕业设计、创新设计水平，全面改变目前大学生动手能力弱的现状，培养当代大学生的创新意识和实际动手能力，提高学生就业竞争能力，并为全国高校间的学术交流提供一个良好的平台。此外，毕昇杯全国电子创新设计竞赛宗旨与“达盛大学计划”旨在为中国大学生建立电子竞赛创新设计的学习环境，遥相呼应，共同为中国培养创新型人才而努力。

2010 年“毕昇杯”全国电子创新设计竞赛于 2009 年 11 月 21 日在北京理工大学国际教育交流大厦三楼报告厅举行了首场竞赛启动发布会，会上主办单位重点说明了以下两点参赛方式：

① 必须在达盛科技开放性设计竞赛平台——E-TRY 电子竞赛创新设计套件基础上，完成创新题目。

② 组委会不对参赛队收取报名费、参赛费、专家评审费等费用。

“毕昇杯”全国电子创新设计竞赛平台采用 E-TRY 电子竞赛创新设计套件，此套件主要包括 CPU 开发板若干系列、通用板和适配器，具体如下：

① CPU 开发板：单片机系列——E-PLAY-51、EXP-C51、EXP-MSP430；DSP 系列——EXP-2407、2812、5402、5509，Techv-2407、2812、28335、5402、5509、6416、6713、6720、6726、6727；ARM 系列——Techv-44B0、2410、PXA270、DM355/335；SOPC 系列——EXP-1K30、1K100、EPM3128、EPM 3256、E-PLAY-1C06、1C12、2C35、3C5、3C10。

② 通用板：与上述三种接口 CPU 开发板配合使用的 TECHV/EXP-II、E-PLAY/EXP-II、EXP/EXP -II 通用板，该通用板可以自由设计，CPU 板可以选择单 CPU 或者双 CPU。

③ 适配器：E-TRY 电子竞赛创新设计套件包含 19 种不同封装芯片用适配器板，帮助设计者将贴片芯片迅速转变为双列直插芯片用于通用板中，更加灵活、方便。

2.3 电子设计竞赛指导

2.3.1 赛前技能训练

对于本科院校来讲，参赛的学生一般是大三学生，为了在竞赛中取得较好成绩，从技能训练安排方面，有些课程如单片机、可编程逻辑器件、传感与检测技术等需要提前学习。原则上，应该提前一个学年开始参赛的技能培训，可以采用如下训练计划。

（1）理论培训

用16周的星期六与星期日，每周 8 个学时，总计约需要培训 16×8=128 学时。培训以单片机、电路设计、仿真软件和传感器等相关学科理论知识为重点。

其中，原则上要进行单片机基础与编程培训 40 学时，可编程逻辑器件与编程 40 学时，Protel 电路设计软件 12 学时，EWB/Multisim 电路设计仿真软件 12 学时，传感与检测技术 24 学时。单片机和可编程逻辑器件课程按基础知识和实际编程两部分进行，以具体的应用为导向，对理论知识不作全面展开，特别是单片机指令和 VHDL 语言，在训练时，应以实际编程为准，做到少而精，特别注重软件开发工具的使用、接口电路的学习与训练。传感与检测技术按必需和够用的原则来安排课程内容，尽可能利用现有的实验设备多做实验。

（2）实践课培训

第 2 个学期进行实践课程培训，也用 16 周的星期六和星期日，每周 8 个学时，16×8=128 学时。按照基础训练、仪器仪表使用、单元电路训练、单片机系统训练、可编程逻辑器件系统训练一步一步进行，以实际动手能力的训练为主。

基础训练主要训练装配工具及使用方法、装配工艺、元器件识别、印制电路板设计与制作等内容。仪器仪表使用主要训练数字万用表、示波器、逻辑分析仪、LC 测试仪、频谱分析仪、信号发生器等仪器仪表的使用。单元电路训练包括电源电路、基本放大电路、射频功率放大器电路、正弦波发生器电路、非正弦波发生器电路、比较器电路、限幅器电路、检波电路、电压/频率变换电路、电压/电流变换电路、光电、红外、超声、金属传感器电路、电机驱动电路、LED 和 LCD 显示与驱动电路、A/D 电路、D/A 电路等。所有电路训练都要求按照工作原理、电原理图、印制板图、装配图、电路调试进行实际制作和整理资料（含设计报告）。单片机和可编程逻辑器件系统训练包括最小系统设计制作、AD/DA 接口电路与程序设计、LED 数码管接口电路与程序设计、LCD 接口电路与程序设计、其他接口电路与程序设计、以及编程技巧、容易混淆的几个概念、常见错误类型及原因分析等内容。所有训练都要求按照工作原理、电原理图、印制板图、装配图、电路调试、程序设计进行实际制作和整理资料（含设计报告）。

（3）强化训练

正式比赛之前的强化训练一般安排在暑假中进行，按 6 周进行。训练目标是模拟实际竞赛。以历年的竞赛题目为模板，适当进行调整。每周一个题目，全部训练要求按照实际竞赛要求进行，竞赛采用“半封闭、相对集中”的组织方式进行。竞赛期间，学生可以查阅有关文献资料，队内学生集体商讨设计思想，确定设计方案，分工负责、团结协作，以队为基本单位独立完成竞赛任务；不允许任何教师或其他人员进行任何形式的指导或引导；参赛队员不得与队外任何人员讨论商量。每次制作完成后，分软件编程、硬件制作、设计总结报告三部分进行分析比较和交流，找出存在的问题。

强化训练可以根据本校参赛学生的实际情况进行，还需要提前考虑一些在竞赛中可能会面临的实际问题，如实际参赛时经费如何解决、竞赛选题方向、竞赛选题是全面展开还是集中在某几个方向，等等。

在训练的第 2 阶段的后半部分，即单片机和可编程逻辑器件系统训练阶段，组织学生进行分组，分组原则按自愿组合进行。每组三人，按照软件编程、硬件制作、设计总结报告写作三部分进行分工，每个队员各有侧重，分工合作。

总体而言，参加电子设计竞赛，如果想在电子设计类的竞赛中获得比较理想的成绩，必须通过赛前的理论培训和实践技能强化训练，熟练地掌握下列各项竞赛技能：

① 能熟练运用 PCB Layer Out 规则（EDA 工具 Protel 99SE，或 AutoCAD）。

② 能熟练进行基于 VHDL、AHDL 的 CPLD、FPGA、GAL 的内核设计。

③ 能熟练进行基于 Multisim 的电路仿真分析。

④ 能熟练地设计开发基于 MCS-51 或其他系列的单片机程序（尤其是基于 C/A 的混合编程）。

⑤ 能熟练进行基于 ASIC 的中小规模时序及组合数字电路设计。

⑥ 能熟练进行基于 ASIC 的通用模拟及高频通信电路设计。

⑦ 能熟练进行基于 ASIC 的 DA/AD 及传感器检测电路设计。

⑧ 能熟练进行基于 ASIC 的锁相环电路及近代频率合成技术开发。

⑨ 能熟练进行单片机的外围扩展电路设计及掌握 MCU 标准通信协议。

⑩ 掌握基于 VB/VC++的上位机程序设计（主要是串并口通信）。

⑪ 能熟练地使用各种通用电参量的定义及测量方法。

⑫ 能熟练地使用万用表、示波器、扫频仪、信号源、频率计等仪表。

⑬ 能在规定时间内独立完成业余条件下的 PCB 设计及制作。

⑭ 能对小规模系统独立完成整机调试。

⑮ 能读懂器件 Data Sheet，并能规范地书写实验手记和设计报告。

⑯ 具备较强的抽象建模能力、较扎实的数学计算能力和英语表达能力，而且谦虚谨慎，具有团队协作精神，还应服从团队决定和任务安排。

2.3.2 比赛过程中的注意事项

1．遵守竞赛纪律

电子设计竞赛采用全国统一命题、分赛区组织的方式，竞赛采用“半封闭、相对集中”的组织方式进行。

下面以 2009 年第 9 届全国电子设计大赛为例介绍。

2009 年 9 月 2 日 8:00，竞赛正式开始，根据自己的技术特长，每支参赛队限定在组委会提供的编号为 A～I 共 9 个命题中任选一题。认真填写《登记表》各栏目内容，填写好的《登记表》由赛场巡视员暂时保存。参赛者必须是有正式学籍的全日制在校本科、专科学生，应出示能够证明参赛者学生身份的有效证件（如学生证）随时备查。每队严格限制三人，开赛后不得中途更换队员。

竞赛期间，可使用各种图书资料和网络资源，但不得在赛区责任学校指定竞赛场地之外的其他任何场所进行设计制作，不得以任何方式与他人交流，包括教师在内的非参赛队员必须回避，对违纪参赛队取消评审资格。

2009 年 9 月 5 日 20:00，竞赛结束，同时上交设计报告、制作实物及《登记表》，由专人封存。竞赛开始后，为保证竞赛工作顺利进行，参赛学校应组织专人负责提供竞赛所需设备、元器件、后勤保障等工作。

2．题目分析

竞赛题目在统一的竞赛开始时间（8:00）打开。以 2009 年第 8 届为例，共有 9 题：光伏

并网发电模拟装置（A 题）、声音导引系统（B 题）、宽带直流放大器（C 题）、无线环境监测模拟装置（D 题）、电能收集充电器（E 题）、数字幅频均衡功率放大器（F 题）、低频功率放大器（G 题）、LED 点阵书写显示屏（H 题）、模拟路灯控制系统（I 题）。其中，本科学生组只能在 A～F 中任选一题，高职高专组可以在全部 9 个题中任选一题进行参赛。

例如，宽带直流放大器（本科组 C 题）。

一、任务

设计并制作一个宽带直流放大器及所用的直流稳压电源。

二、要求

1．基本要求

（1）电压增益 $A_V \geqslant 40$ dB，输入电压有效值 $V_i \leqslant 20$ mV。A_V 可在 0～40 dB 范围内手动连续调节。

（2）最大输出电压正弦波有效值 $V_o \geqslant 2$ V，输出信号波形无明显失真。

（3）3 dB 通频带 0～5 MHz；在 0～4 MHz 通频带内，增益起伏≤1 dB。

（4）放大器的输入电阻≥50 Ω，负载电阻 50 Ω±2 Ω。

（5）设计并制作满足放大器要求所用的直流稳压电源。

2．发挥部分

（1）最大电压增益 $A_V \geqslant 60$ dB，输入电压有效值 $V_i \leqslant 10$ mV。

（2）在 $A_V = 60$ dB 时，输出端噪声电压的峰－峰值 $V_{ONPP} \leqslant 0.3$ V。

（3）3 dB 通频带 0～10 MHz；在 0～9 MHz 通频带内，增益起伏≤1 dB。

（4）最大输出电压正弦波有效值 $V_o \geqslant 10$ V，输出信号波形无明显失真。

（5）进一步降低输入电压提高放大器的电压增益。

（6）电压增益 A_V 可预置并显示，预置范围为 0～60 dB，步距为 5 dB（也可以连续调节）；放大器的带宽可预置并显示（至少 5 MHz、10 MHz 两点）。

（7）降低放大器的制作成本，提高电源效率。

（8）其他（如改善放大器性能的其他措施等）。

三、说明

（1）宽带直流放大器幅频特性示意图，如图2.1 所示。

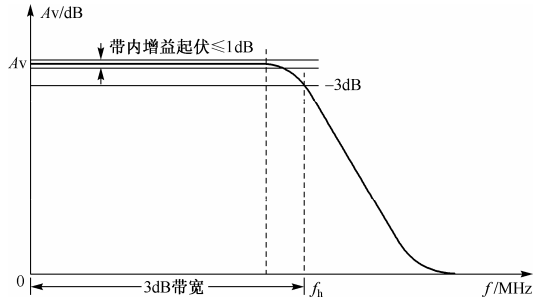


图 2.1　幅频特性示意图

（2）负载电阻应预留测试用检测口和明显标志，如不符合 50 Ω±2 Ω的电阻值要求，则酌情扣除最大输出电压有效值项的所得分数。

（3）放大器要留有必要的测试点。建议的测试框图如图2.2所示，可采用信号发生器与示波器/交、直流电压表组合的静态法或扫频仪进行幅频特性测量。

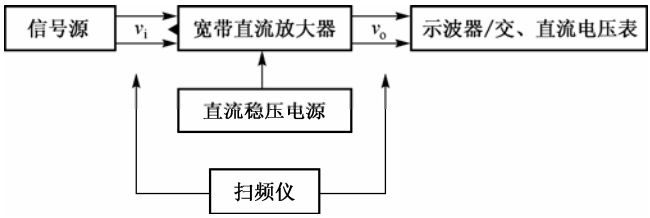


图 2.2　幅频特性测试框图

四、评分标准

	项　目	主要内容	分数
设计报告	系统方案	比较与选择\方案描述	2
	理论分析与计算	带宽增益积\通频带内增益起伏控制\线性相位\抑制直流零点漂移\放大器稳定性	9
	电路与程序设计	电路设计	8
	测试方案与测试结果	测试方案及测试条件\测试结果完整性\测试结果分析	8
	设计报告结构及规范性	摘要\设计报告正文的结构\图表的规范性	3
	总分		30
基本要求	实际制作完成情况		50
发挥部分	完成第（1）项		7
	完成第（2）项		2
	完成第（3）项		7
	完成第（4）项		6
	完成第（5）项		12
	完成第（6）项		5
	完成第（7）项		6
	其他		5
	总分		50

参赛队员应仔细阅读所有的竞赛题目，根据自己组三个队员各自的技术专长和强化训练情况，选择相应的题目作为参赛题目。选择题目时，应该仔细阅读题目说明文件（如上例），弄清楚题目的任务和要求，还应该阅读该题目的评分标准细则。在选题时，应注意题目中不应该有知识盲点，要能够完全看懂题目的任务、准确无误地理解其要求。如果不能完全看懂题目的任务和要求，原则上不要盲目选择该题目。此外，竞赛时间安排非常紧凑，根据竞赛纪律规定，选题时不可以去请教老师，否则视为违纪并取消资格。

设计制作分基本要求和发挥两大部分，每部分各占 50 分，设计与总结报告 30 分（专科组 20 分），因此竞赛总分为 130 分（专科组 120）。注意，基本部分的各项分值题目中是没有给出的，只在发挥部分给出了各小项的具体分值，因此在分析题目、选择确定参赛题目时，要仔细阅读分析各项要求细则，做到重点突出、技术指标符合原题要求。

通读各题说明文件后，对各题目进行分析和对比，再确定最终的参赛题目，确定竞赛题

目的总时间不要超过 2 小时，题目一旦选定，原则上应保证不要中途更改题目，否则会影响比赛信心和后期任务进程。

3．方案的设计

确定好竞赛题目后，参赛小组的三个队员应认真地讨论设计方案，设计方案应考虑制作的可行性，包括元器件是否齐备、是否能够采购到等，都必须考虑清楚。初步的设计方案可以提出1～2个，或者确定一个方案，再对该方案中的某些部分做出两个或者两个以上的改进方案。例如，宽带直流放大器（C 题）在总体方案确定的情况下，可以提出分别采用单片机或者可编程逻辑器件作为主控制器。因此配套的接口元器件就会有所不同，再进一步综合论证，最终是采用单片机还是采用可编程逻辑器件作为主控制器。在论证时，要结合题目的技术指标要求、队员的技术专长、可供选用的元器件等因素综合考虑。

确定技术方案以后，应尽快列出所需电子元器件的清单，确定各元器件类别、数量是否齐备，对一些方案中需要但在竞赛组委会公布的“基本仪器和主要元器件清单”中尚未备齐的要迅速提出采购（或租借）清单。元器件采购（或租借）清单应完整提供元器件的名称、型号及规格，如电阻器 RJ-14—0.25W—510 Ω ± 5%、所需数量、可用的替代型号及规格。

从时间上考虑，确定技术方案的时间不应该超过 6 小时，否则会延误比赛进程，占用后续任务的工作时间，导致比赛成绩不理想。

4．元器件的采购（租借）

元器件采购（租借）是保证竞赛顺利进行的基础，没有元器件，参赛作品是不可能制作成功的。因为竞赛题目事先是不可能知道的，学生做出的设计方案也是各有不同，需要的元器件也会是多种多样的。尽管各参赛队可以事先准备一些元器件，但不可能完全备齐。

在竞赛中进行临时性的元器件采购（租借）是经常会遇见的。不同的院校处在不同的城市，元器件的供应情况是不同的。以电子科技大学为例，要保证能够采购（租借）到所有竞赛需要的元器件，比较保险的做法是到电子市场去采购（租借）。因此，在竞赛开始前两天，派 1～2 名老师先去联系相关电子元器件供应商。竞赛开始后，竞赛学生不断地将需要的元器件清单发送给联系教师，由联系教师临时去租借或购买。

值得注意的是，应当要求学生应尽快拿出元器件清单，以便及时采购（租借），避免因元器件缺乏影响比赛进程。可以让学生分多次提出，联系教师多批次购买（租借），确保各队所需元器件能及时交付使用。

5．参赛作品的制作与装配

在竞赛中，各队学生可以按照在训练中的分工，按照软件编程、硬件制作、设计总结报告写作三部分分头进行，每个队员各有侧重，注意分工合作。

设计制作过程中，可以分模块设计、安装、调试，一步一步地进行。三人之间要注意多沟通，多交流，遇到问题，不要慌张，要冷静处理。在设计、安装、调试的每一步，都需要两人以上进行核对和检查。特别是总装调试阶段，一定要格外小心，通电前一定要两人以上进行核对和检查。

参赛作品应该在第 4 天的下午全部完成。根据多次竞赛总结的经验，到第4天的下午没有完成的功能和指标，也就不要再做了。很有可能的是，为了追求更好的指标，或者更完全的功能，结果将已经完成的部分给弄坏了。几乎在每次竞赛中都有这样的参赛组，在最后一步前功尽弃。

负责设计总结报告写作的队员，要根据软件和硬件设计队员提供的电路设计图纸、程序清单等设计资料，按照设计总结报告写作要求进行整理。到第 4 天的上午，设计总结报告应该全部完成，有省略的应该是测试数据部分，因为作品此时可能还在调试过程中。测试数据部分可以在作品调试完毕，竞赛结束前，补充到设计总结报告中去。在作品和设计总结报告中，一定不能出现参赛队所属学校名称、参赛队员和指导教师的姓名以及相关标记。

在开赛后第4天晚上 20:00，竞赛全部结束，此时各队需要上交设计报告、制作实物及《登记表》，由赛场巡视员封存。封存时应注意文件完整，作品要妥善包装。包装要防震，要能够防止运输和搬运过程的冲击和震动。

6．其他注意事项

整个竞赛时间将持续4个白天3个晚上，时间跨度较长。在竞赛过程中一定要安排好休息时间。要求竞赛队员要保证一定的休息时间，特别是第 1 天和第 2 天。不要第1天和第 2 天不休息，到了第 3 天和第 4 天，队员十分疲劳，昏头昏脑，效率将十分低，也容易出错。在 9 月，许多地方白天的天气还是很热，而晚上的天气则较凉快。队员在休息时注意不要着凉。

在4天的竞赛期间，参赛队员精神高度紧张，身体也容易疲劳，通常饮食胃口不会太好。参赛学校应组织专业人员负责提供饮食及后勤保障，最好能根据参赛队员的不同要求，为参赛队伍提供干净、营养的餐饮服务。

2.3.3 参赛经验交流

1．对竞赛满怀热情

热情是一个人保持对一件事物的热衷程度，可以引导你，为你注入强大的动力。有人参加竞赛纯粹是为了将来找工作时可以在自己的简历上填上这个经历，这就不是一种热情了。因为对他们来说，参加竞赛才是首要的，至于能否拿奖次之，也就不注重竞赛的过程。这不值得提倡，应提倡的是用热情参赛，但不提倡三分钟的热情。其次，如果面临考研和参赛的抉择，最好不要迟疑太久，要明确自己的方向，当机立断，免得两头分心，吃力又不合算（毕竟一个人精力有限）。正所谓，好的开始就是成功的一半。

2．关注和收集比赛资料，了解规则

如果你有意参加的话，平时就要多关注全国电子竞赛信息，收集一些往年竞赛资料，了解评分规则，这些都是很有必要的。

3．团队的选择

一个能在全国大学生电子竞赛中得奖的团队必须具备如下特征：绝对喜欢电子开发；在软

件、硬件、论文写作中各有所长。你想想，三个只会 C 语言的家伙在一起能干什么。这样的团队还不如大家一起看看《越狱》、《24 小时》实在。选好团队后就要确定团队核心人物。大家一定都能想得到，一旦进入备战阶段，组员间的探讨是必然的，各抒己见固然好，也往往比较容易起争执。这时如果有个核心人物，他的话在非常时期大家必须绝对服从，他说了算。

4．正确认识自己的团队优势资源，明确分工

要想获得成功，就必须认清自己团队的优势，充分调用组员中的优势资源，比如：某个组员在编程方面有优势，那么就应该合理分配编程工作给他，这样大家分工也就相对明确了，也都能充分发挥自己的所长，将团对协作的力量发挥到最大。

5．基本技能的准备

磨刀不误砍柴工，根基不牢就不要往上爬，爬得越高，摔得越重。需要学会连接常用的电路，如高增益放大电路、跟随电路、滤波电路，学习 Protel 制图软件，会画电路原理图，会 PCB 制版。在准备期间，一定要学会用万用板焊接电路，或者用三氯化铁腐蚀电路，这个效果非常好。在正式比赛前，一定要与团队在4天内练习做一个完整的电子系统，时间安排在8月初最好（推荐凌阳大学计划赛前热身套件）。不要以为你能做好各个模块，联机调试就一定能成功，总是会缺点什么。事实证明，大部分的竞赛队失败就在最后的联机。

6．选对核心器件是关键

（单片机）选型 CPU 是电子设计竞赛的核心器件。微处理器很多，如比较古老的 8051 单片机、AVR 单片机、PIC 单片机、MSP430 单片机。现在也有很多新型单片机，这类单片机含有很多扩充资源，如大量的 FLASH（这些 FLASH 对 LCD 字库很有帮助）、中速 AD、D\A、多定时器、PWM，语音功能等，这些资源会给开发带来很多方便。从个人经验看，竞赛一定要用自己擅长的单片机，但根据题目的不同，不同的单片机实现起来难度也不一样。如果准备时间短，没有很丰富的单片机开发经验又对自己水平不是太自信的朋友，可以学习一下凌阳的61板，快速入门和丰富权威的自学资料能达到事半功倍的效果。随着时代的变化，电子产品的更新也是日新月异的，现在很多人都喜欢上了嵌入式和 DSP。根据 2009 年竞赛的精神，利用嵌入式和 DSP 参加电子竞赛也会逐渐提到日程上来，如果做的是关于高速数据处理的题目，如高速数据采集、高速数据传输等，嵌入式和 DSP 就会有用武之地。

7．基本模块的准备

参加电子设计竞赛，离不开一些基本模块，如 LED、LCD 子电路、子程序（这点凌阳的 61 板做得很好，有很多现成的标准函数可以调用）。ADDA 电路，搞控制的总得选好步进电机的型号、驱动电路、驱动程序吧。那搞无线的应该准备什么呢？自己去想吧！有一个一等奖选手告诉过我，他说：“在竞赛前我就知道我会得奖，因为我把该准备的东西都准备了，而且在比赛前几天都梦到了比赛题目。”他说得有点夸张，但是也很可信，足以证明准备对整个比赛有举足轻重的作用。

8．专注自己的强项

下面谈几点全国大学生电子设计竞赛的试题情况。其实每年的竞赛试题都有很多相似的地方，控制类的有简单的工业控制、小车；传统题目有数据调理、数据采集、无线传输系统；电源设计有简单仪器、仪表设计（每年必有一题），如 2005 年的简易频谱分析仪，2003 年的简易逻辑分析器。建议期望夺奖的朋友专注自己的强项。

9．备战前要调理好身体

这也是很容易忽略的一点。4 天 3 夜是什么概念，我想大家都清楚，所谓身体是革命的本钱。既然下定决心参赛，就要调理好身体，以最佳的状态来迎接挑战。否则，千载难逢的机会就可能擦肩而过，留下的将会是遗憾。

10．要有善始善终、持之以恒的决心

竞赛不仅仅是技能的竞争，更是良好心态的比拼。比赛期间，肯定会遇到这样或那样的问题，如果没有善始善终、持之以恒的决心，三天打鱼，两天晒网，是走不到最后的。要切切实实按照制定好的进程走，该做哪项任务，就要尽量克服一切困难做完，该今天做的，就不能留到明天。

具备以上条件后，你还需要的就是自信了。

11．会做也要会表达

一个获得一等奖的团队，不用问这里面肯定有一个好的队长，一个好的指导老师，还有一个十分关键的角色是文笔不错且善于表达的队员。一个团队的设计理念和思想需要通过他公布于众，让评委们了解和认同其观点，论文的书写也是考察写作基本功的关键要素，切记现在不是酒香不怕巷子深的年代了，因为可以选择的对象太多，要学会主动出击，一个好的团队是成功的一半。

2.4 相关技术交流网站

全国大学生电子设计竞赛的官方网站是 http://www.nuedc.com.cn，所有与竞赛相关的信息都可以在这个网站上查询到。

除此之外，还有一些相关的技术交流网站需要关注。

1．http://www.nuedc.org/

大学生电子设计联盟网，号称是中国最大、最专业的大学生电子设计交流社区专业平台。该网站提供了专业的电子设计竞赛交流论坛，可以与历届电子竞赛获奖者交流参赛经验，还可以与同行交流单片机技术和其他综合电子技术，是参加各类电子设计竞赛的学习和交流理想家园。该网站倡导的宗旨是：互相学习，共同进步，资源共享，倡导开源。

2．http://bbs.21ic.com/

这个网站是个老牌的专业电子论坛，其专业性、人气在国内首屈一指。

3．http://community.csdn.net/

这个网站也是个老牌的专业程序设计论坛。现在的电子产品不再是只靠 MCU 就能行的，微机会扮演愈来愈重要的角色，如 RS-232 通信、USB 接口、数据分析（利用 MCU 采集，微机进行数据分析）等。有微机就需要编程，编程就需要释疑，这里值得尝试。

4．http://edc.saicai.com/bbs

这个网站里有很多很好的第一手电子设计资料，也是最新电子设计领域赛事的发布平台，同时聚集了大量的电子设计爱好者参与讨论。

5．http://www.eetchina.com/

这个网站提供了有关电子工程及电子设计的最新资讯和科技趋势。从事电子设计工作不能闭门造车，需要了解最新的领域行情，从这个网站可以了解到电子设计行业的重要资讯。

6．http://www.icbuy.com/

这是一个电子元器件的交易平台网站，能够很好地进行芯片选型，还能收到该网站发送的免费产品目录。

7．http://www.edaboard.com/

这是一个国际知名的电子行业论坛，类似 bbs.21ic.com。假如你有疑问，而且英文还过得去，可以在这里浏览，可以解决不少技术疑难问题，而且外国同行一般很热情。在这个网站上，既可以解决电子设计中遇到的技术问题，还可以顺便提高英语水平。

8．http://www.wikipedia.org/

这个网站通常叫做“维基百科”，是一个知识搜索网站，可以搜索到大多数的电子领域专业名词，最重要的是，从这些被搜索的名词条目出发，可以顺便找到很多相关的知识。如搜索 MSP430，可以找到 Texas Instruments. PWM、watchdog、USART、SPI、I2C ADCs、DAC、LCD op-amps EPROM JTAG MIPS、RS-232 等，从这些词又可以找到其他的，能构建一个复杂的知识网。

9．http://www.datasheetdir.com/

Datasheet Directory，也叫芯片目录网。在这个网站中不需要下载 PDF 文件就可以看到芯片的引脚分布和基于此芯片的应用电路。

10．http://www.eetimes.com/

这是一个类似 http://www.eetchina.com/的技术网站，提供丰富的电子领域英文资讯。

11．http://makezine.com/

国外的 DIY 网站，其中有很多电子项目，从这里可以看到很多有趣的项目，可以尝试去动手开发这些项目，对提高电子设计实践能力能起到事半功倍的作用。

第 3 章　主教材的习题解答

3.1　第 1 章习题解答及解题思路

3.1.1　第 1 章重点和难点解析

本章的重点和难点主要包括冯・诺依曼思想、信息的数字化表示和存储程序工作方式。下面对这几个问题加以说明。

1．冯・诺依曼思想

关于冯・诺依曼思想，首先要了解它是因解决什么问题而产生的，然后应清楚该思想的要点是什么，最后需知道该思想对计算机发展所产生的深远意义及改进的方向。

冯・诺依曼思想是为解决信息如何表示才能被计算机识别和采用什么工作方式，才能使计算机自动对信息进行处理这两个基本问题而产生的。

为此，冯・诺依曼思想包含三个要点：① 采用二进制代码形式表示程序和数据；② 采用存储程序工作方式；③ 计算机硬件系统由存储器、运算器、控制器、输入设备和输出设备等部件组成。第一个要点说明信息的表示要数字化；第二个要点说明计算机应采用存储程序的工作方式，这是冯・诺依曼思想最核心的概念；第三个要点说明要达到上述目标需提供相应的硬件支持：输入设备输入程序、数据等信息，存储器存储信息，运算器对数据进行加工处理，输出设备输出处理结果，控制器执行由程序转换成的指令序列对整个过程进行控制。

冯・诺依曼思想提供了数字计算机的基本设计思想，奠定了现代计算机的基本结构，并开创了程序设计的新时代。传统的诺依曼机采用串行处理的工作机制，因此要想提高计算机的性能，其方向之一是采取并行处理机制。

2．信息的数字化表示

我们讨论的是数字计算机，因此要从计算机的物理机制和对原始信息的表示两方面来理解信息的数字化表示。由于计算机是由电子部件组成，处理的是电平或脉冲这样只具有两种状态的数字型电信号，如电平的高或低、脉冲的有或无，因此要让计算机能够对各种各样的原始信息进行处理，就需要首先将原始信息表示为只取两种状态值的数字代码（二进制代码），这是信息表示数字化的第一层含义。需要注意的是，一位二进制代码只有 0、1 两种状态，而多位二进制代码的组合则有多种状态，可以表示多个信息。例如，3 位二进制代码有 8 种状态，可以表示 8 个数据，或者 8 个命令，或者 8 种颜色等。然后，从物理实现的角度，将数字代码表示为数字信号（电平或脉冲），才能被计算机直接处理，这是信息表示数字化的第二层含义。

了解信息数字化表示的含义，并善于用约定的数字代码表示各种需要描述的信息，这是理解计算机工作原理的一个基本出发点，也是从事计算机技术工作的重要前提。

3．存储程序工作方式

要让计算机自动、连续地工作，就需要告诉计算机做什么和怎样做，这是通过存储程序方式来实现的。存储程序工作方式包含三个要点：① 根据求解的问题事先编制程序；② 将编好的程序先存入计算机中；③ 计算机自动、连续地读取并执行程序。注意，用程序设计语言编写的源程序在被送入计算机后，仅仅是被转换成了二进制代码形式，还不能被计算机识别和执行，需要再经过编译或解释，转换为按专门指令格式组成的指令序列，即目标程序，并存放在存储器中，才能被计算机读取和执行。

3.1.2 第 1 章习题解答与解题思路

1．简要解释下列名词术语

【答】

数字计算机：一种能存储程序，能自动、连续地对各种数字化信息进行处理的快速工具。

硬件：指组成计算机系统的设备实体，如 CPU、存储器、I/O 设备等。

软件：泛指各类程序、文档等。

CPU：即中央处理器，是由运算器和控制器组成的计算机硬件系统的核心部件。

主存储器：位于主机内部，用来存放 CPU 需要使用的程序和数据的部件。

外存储器：位于主机外部，用来存放大量的需要联机保存、但 CPU 暂不使用的程序和数据的部件。

外部设备：位于主机之外，与主机进行信息交换的输入设备或输出设备。

信息的数字化表示：包含两层含义，即用数字代码表示各种信息、用数字信号（电平、脉冲）表示数字代码。

存储程序工作方式：事先编制程序，事先存储程序，自动、连续地执行程序。

模拟信号：在时间上连续变化的电信号，用信号的某些参数模拟信息。

数字信号：在时间上或空间上断续变化的电信号，依靠彼此离散的多位信号的组合表示信息。

脉冲信号：在时间上离散的电信号，利用脉冲的有、无表示不同的状态。

电平信号：在空间上离散的电信号，利用信号电平的高、低表示不同的状态。

系统软件：为保证计算机系统能够良好运行而设置的基础软件。

应用软件：用户在各自的应用领域中为解决各类问题而编写的软件。

操作系统：负责管理和控制计算机系统的硬件资源、软件资源和运行的核心软件，为用户提供软件的开发环境和运行环境。

语言处理程序：将源程序转换为目标程序的一类系统软件，包括各种解释程序、编译程序、汇编程序。

物理机：指能够执行机器语言程序的实际计算机。

虚拟机：指通过配置软件，扩充机器功能后所形成的计算机。

总线：一组能为多个部件分时共享的公共的信息传送线路。

数据通路宽度：指数据总线一次能并行传送的数据位数。

数据传输速率：指数据总线每秒传送的数据量。

接口：泛指两个部件的交接部分。

通道：能够执行专用的通道指令，用来管理 I/O 操作的控制部件。

字节：8 位二进制代码称为 1 字节。

字长：一般指参加一次定点运算的操作数的位数。

2．数字计算机的主要特点是什么？

【答】

应该从信息的表示方法和计算机的工作方式来说明它的主要特点。

数字计算机的主要特点包括：能自动连续地执行程序、运算速度快、运算精度高、存储能力强、通用性好。

3．计算机有哪些主要性能指标？

【答】

主要从计算机的运算能力、存储能力、传送能力、处理能力等方面考虑。

主要性能指标包括基本字长、运算速度、存储容量（主存容量和外存容量）、数据传输速率、外设配置和软件配置等。

4．冯·诺依曼思想包含哪些要点？

【答】

冯·诺依曼思想奠定了现代计算机的基本结构思想，很好地解决了信息如何表示才能被计算机识别和计算机采用何种工作方式才能自动对信息进行处理等基本问题，包含三个要点：

① 采用二进制代码表示信息，以便计算机识别；

② 采用存储程序工作方式，才能使计算机自动地对信息进行处理；

③ 由存储器、运算器、控制器、输入/输出设备等功能部件组成计算机硬件系统。

5．信息的数字化表示包含哪两层含义？

【答】

信息的数字化表示不仅要考虑在计算机中如何表示各种原始信息，还要考虑在物理机制上怎样实现。所以，第一层含义为：用数字代码表示各种信息；第二层含义为：用数字信号表示数字代码。

6．用数字信号表示代码有什么优点？

【答】

每位数字信号只有两种可能的状态，因而可从物理实现、可靠性、数值范围与精度、信息类型、信息处理等方面说明，有以下 5 点：

① 在物理上容易实现信息的表示与存储；

② 抗干扰能力强，可靠性高。

③ 数值的表示范围大，表示精度高。

④ 能表示极其广泛的信息类型。

⑤ 能用数字逻辑技术处理信息。

7．编译方式和解释方式对源程序的处理有什么区别？

【答】

在编译方式中，计算机执行编译程序，将源程序全部转换为目标程序，然后由计算机单独执行目标程序，即先翻译后执行。

在解释方式中，计算机执行解释程序，将源程序逐段转换为对应的目标程序段，每转换一段，便执行该段目标程序，直到整个源程序被解释执行完，即边翻译边执行。

8．为什么要对计算机系统进行层次划分？

【答】

计算机系统是由硬件、软件组成的复杂系统，进行层次划分，有助于根据不同需要，从不同层次去分析、构造、调试、维护和扩充计算机系统。

9．软件系统一般包含哪些部分？试列出你所熟悉的几种系统软件。

【答】

前面几道题都是涉及基本概念的题，从这道题开始，则是与实际应用有关。

软件系统一般包含系统软件和应用软件两部分。所熟悉的系统软件可根据实际情况列出，如操作系统（Windows、Linux 等）、C 编译程序、数据库管理系统（SQL Server、Sybase 等）。

10．以你所熟悉的一种计算机系统为例，列举出该系统所用的CPU型号、时钟频率、字长、主存容量、外存容量、所连I/O设备的名称等。

【答】

例如使用奔腾芯片的计算机系统，CPU 为 Pentium-200，时钟频率为 200 MHz，字长 32 位，主存容量为 256 MB，硬盘容量为 40 GB，I/O 设备包括键盘、鼠标、显示器、喷墨打印机等。

11．什么是控制流驱动？什么是数据流驱动？

【答】

传统的冯·诺依曼机采用控制流（指令流）驱动方式，按指令序列依次读取指令，根据指令所包含的控制信息对数据进行处理，在程序执行过程中，始终由指令流驱动计算机工作。

数据流驱动方式是对传统冯·诺依曼机工作方式的根本改变：只要数据准备好，有关指令就可并行执行，如数据流计算机。

12．你曾在计算机的机器指令级、操作系统级、汇编语言级或高级语言级上做过工作或练习，或调用过该级的功能吗？举出所做的工作或所调用的功能名。

【答】

按实际情况回答，如用汇编语言或高级语言编写过程序等。

3.2 第 2 章习题解答及解题思路

3.2.1 第 2 章重点和难点解析

1．二进制、八进制、十六进制数之间的转换

（1）二进制数转为八进制数

方法：找到小数点位置，向左每三位一组（高位不够补 0），每组用一位八进制数码表示，则获得八进制数的整数部分；从小数点位置向右每三位一组（低位不够补 0），每组用一位八进制数码表示，则获得八进制的小数部分，两部分合起来则是该二进制数的八进制表示。

例如，$(10110011.10110)_2=(263.54)_8$。

（2）八进制数转为二进制数

方法：将每位八进制数码均用相应的三位二进制数表示，若最高位（或最低位）是 0（包括与它相邻的 0）可省去。

例如，$(134.76)_8=(1011100.11111)_2$。

（3）二进制数和十六进制数的转换

方法：同二进制数和八进制数的转换方法类似，不同之处是，二进制数是按每四位一组进行分组，每组用一个十六进制数码表示，则获得十六进制的数；将十六进制数的每个数码用相应的四位二进制数表示，则得到该数的二进制表示。

例如，$(10101101011.1110101)_2=(56B.EA)_{16}$。

（4）八进制数和十六进制数的转换

方法：先将八进制数（或十六进制）数转换为二进制数，再由二进制数转换为十六进制（或八进制）数。

例如，$(74.6)_8=(111100.11)_2=(3C.C)_{16}$

（5）当将 2^n 为分母的分数转换为二进制数时，采用如下方法。

例如，$\left(-\frac{5}{64}\right)_{10}=\left(-\frac{5}{2^6}\right)_{10}=(-5\times2^{-6})_{10}=(-101\times(10)^{-110})_2=(-0.000101)_2$。

2．原码↔补码的转换方法

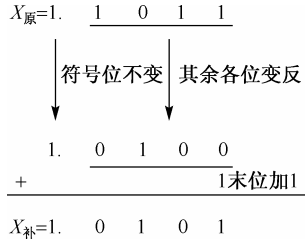
（1）正数的补码表示和原码表示其形式相同

例如，若 $X_{原}=0.1011$，则 $X_{补}=0.1011$。

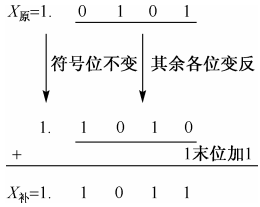
（2）负数的原码↔负数的补码的转换方法

方法一：符号位不变，其余各位变反，末位再加 1。

例如，若 $X_{原}=1.1011$，求 $X_{补}$=?

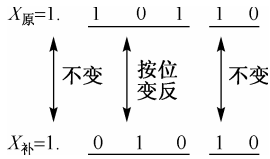


例如，若 $X_{补}$=1.0101，求 $X_{原}$ =?



方法二：符号位不变，尾数部分自低位向高位（看），遇到第一个 1 及其以前的各位 0 都保持不变，以后的各高位按位变反。

例如，$X_{原}$=1.10110，则



3．原码、补码、反码的标识范围和特殊值的机器数表示

（1）表示范围

若真值 X 为 n 位，机器数为 n+1 位（含一位符号），则

定点小数的原码和反码表示范围是：$-1<X<1$ 或 $-(1-2^{-n})\sim(1-2^{-n})$；

定点小数的补码表示范围是：$-1\leqslant X<1$ 或 $-1\sim(1-2^{-n})$；

定点整数的原码和反码表示范围是：$-2^{n}<X<2^{n}$ 或 $-(2^{n}-1)\sim(2^{n}-1)$；

定点整数的补码和移码表示范围是：$-2^{-n}\leqslant X<2^{n}$ 或 $-2^{n}\sim(2^{n}-1)$。

注意：

① 上述结论来源于原码、补码、反码表示的定义式。

② 负数补码表示比原码表示多一种组合，即：定点小数时，补码可表示–1；定点整数时，补码可表示 -2^{n}；原码则不能表示。这一结论来源于负数补码和负数原码表示的定义式，在两者的定义式中，数 x 的取值范围略有不同。

③ 定点小数时，–1 的补码不能按前面讲的方法转换为原码；定点整数时，-2^{n} 的补码也不能按前面讲的方法转换为原码，其原因是在定点小数时原码不能表示 –1；在定点整数时，原码不能表示 -2^{n}。其根本原因是补码和原码表示的定义式（包括 X 取值范围）不一样。

（2）数值 0 的机器数表示

按照原码、补码、反码表示的定义式，在原码和反码表示的定义式中，真值 X 在正值域和负值域中都可以为 0，故将 0 代入定义式中，会得到两个原码（或反码）；而在补码表示的定义式中，真值 X 在正值域中可以为 0，而在负值域中不能为 0，故 0 的补码只有一个。以定点小数为例，数值 0 的原码、反码和补码表示如下：

$$[+0]_{原}=0.00\text{L}\ 00$$

$$[-0]_{原}=1.00\text{L}\ 00$$

$$[+0]_{反}=0.00\text{L}\ 00$$

$$[-0]_{反}=1.11\text{L}\ 11$$

$$[0]_{补}=0.00\text{L}\ 00$$

整数 0 的补码和移码为：$[0]_{补}=000\mathrm{L}\ 00$

$[0]_{移}=100\mathrm{L}\ 00$

4．定点数和浮点数表示

（1）定点数表示

在计算机中，小数点位置固定的数叫做定点数。定点数通常有三种：无符号定点整数、带符号定点小数和带符号定点整数。

① 若某机字长 8 位，含一位数符，采用原码表示，则定点小数所能表示的非零最小正数、最大正数、绝对值最小负数和绝对值最大负数各为多少？该定点小数表示范围是多大？

非零最小正数为 0.0000001，即 2^{-7}，其原码为 0.0000001；

最大正数为 0.1111111，即 $1-2^{-7}$，其原码为 0.1111111；

绝对值最小负数为 –0.0000001，即 -2^{-7}，其原码为 1.0000001；

绝对值最大负数为 –0.1111111，即 $-(1-2^{-7})$，其原码为 1.1111111；

该定点小数表示范围是 $-(1-2^{-7})\sim(1-2^{-7})$。

② 若某机字长 8 位，含一位数符，采用原码表示，则定点整数所能表示的非零最小正数、最大正数、绝对值最小负数和绝对值最大负数各为多少？该定点整数表示范围是多大？

非零最小正数为 1，其原码为 00000001；

最大正数为 1111111，即 2^7-1，其原码为 01111111；

绝对值最小负数为 –1，其原码为 10000001；

绝对值最大负数为–1111111，即 $-(2^7-1)$，其原码为 11111111；

该定点整数表示范围是 $-(2^7-1)\sim(2^7-1)$。

③ 若某机字长 32 位，含一位数符，采用补码表示，则定点小数所能表示的非零最小正数、最大正数、绝对值最小负数和绝对值最大负数各为多少？该定点小数表示范围是多大？

非零最小正数为 0.00…001（小数点后共 30 个 0），即 2^{-31}，

其补码表示为 0.00…001（小数点后 30 个 0）；

最大正数为 0.11…11（小数点后 31 个 1），即 $1-2^{-31}$，

其补码表示为 0.11…11（小数点后 31 个 1）；

绝对值最小负数为–0.00…01（小数点后 30 个 0），即 -2^{-31}，

其补码表示为 1.11…11（小数点后 31 个 1）；

绝对值最大负数为 –1，其补码表示为 1.00…00（小数点后 31 个 0）；

该定点小数表示范围是–1～$(1-2^{-31})$。

④ 若某机字长 32 位，含一位数符，采用补码表示，则定点整数所能表示的非零最小正数、最大正数、绝对值最小负数和绝对值最大负数各为多少？该定点整数表示范围是多大？

非零最小正数为 1，其补码表示为 000…0001（在数码 1 之前共 31 个 0）；

最大正数为 111…11（共 31 个 1），即 $2^{31}-1$，其补码表示为 0111…111（共 31 个 1）；

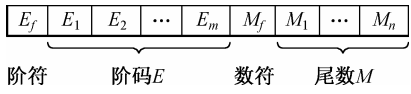
绝对值最小负数为 –1，其补码表示为 111…11（共 32 个 1）；

绝对值最大负数为 -2^{31}，其补码表示为 100…00（共 31 个 0）；

该定点整数表示范围是$-2^{31}\sim(2^{31}-1)$。

（2）浮点数表示

① 浮点数表示的一般格式如下：



浮点数 N 的真值为 $N=\pm R^E\cdot M$ 。

式中，E 和 M 分别是浮点数阶码 E 和尾数 M 的真值；R 是阶码的底，其值与尾数 M 的基数相同，是隐含约定的。

② 引入规格化尾数的目的是什么？规格化尾数的含义和特征是什么？

在浮点数表示中，为了充分利用尾数部分的有效位数，使精度尽可能地高。一般都对尾数采取规格化约定，这样的尾数称为规格化尾数，相应的浮点数是规格化浮点数。

在 R=2（即阶码以 2 为底或尾数采用二进制表示）的情况下，规格化尾数的含义是尾数 M 满足下述条件：

$\frac{1}{2}\leqslant|M|<1$，即尾数的真值 $M_{真}=\pm0.1M_2M_3\mathrm{L}\ M_n$（$M_i$ 为 0 或 1）。

一个浮点数的尾数是否是规格化表示呢？我们可以根据规格化尾数的满足条件提取出规格化尾数的特征来进行判别。

尾数是正数或真值：规格化尾数的特征是 M_1=1；

尾数是负数且原码表示：规格化尾数的特征是 M_1=1；

尾数是负数且补码表示：规格化尾数的特征是 M_1=0（$M_{真}=-\frac{1}{2}$ 是个例外）。

$M_{真}=-\frac{1}{2}$，则 $M_{原}=M_{补}$=1.100…00，是规格化尾数。

作为练习，现假设浮点数的 R=2，其尾数如下，判别哪些是规格化尾数。

$A_{补}$=1.0110001　　$B_{补}$=1.1000000　　$C_{原}$=1.1000000

$D_{原}$=1.0110011　　$E_{补}$=0.0111111　　$F_{补}$=0.1000000

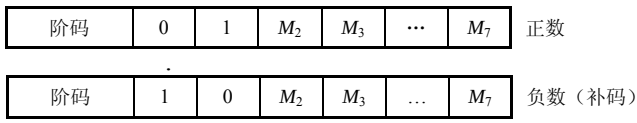
$G_{补}$=00.010111（双符号位）　$H_{补}$=11.001001（双符号位）　$I_{原}$=11.110101（双符号位）

$J_{原}$=00.101011（双符号位）　$K_{补}$=11.101011（双符号位）　$L_{原}$=11.010100（双符号位）

根据规格化尾数的特征可判别出 $A_{补}$、$B_{补}$、$C_{原}$、$F_{补}$、$H_{补}$、$I_{原}$和 $J_{原}$是规格化尾数。相应地，$A_{原}$、$B_{原}$、$C_{补}$、$F_{原}$、$H_{原}$、$I_{补}$和 $J_{补}$也是规格化尾数

③ 若浮点数字长 12 位，其格式如①所示。其中阶码含一位阶符共 4 位，以 2 为底，补码表示；尾数含一位数符共 8 位，补码表示，规格化。则浮点数所能表示的非零最小正数、最大正数、绝对值最小负数和绝对值最大负数各为多少？求该浮点表示范围。

本例的规格化尾数形式如下：



要使浮点数为非零最小正数，则尾数应该是非零最小正数，且是规格化尾数，故为0.1000000（即2^{-1}）；阶码应该是绝对值最大负数 1000（即-2^3）。于是，该浮点数格式所能表示的非零最小正数为

1000	01000000

$2^{-2^3}\times 0.1=2^{-8}\times 2^{-1}=2^{-9}$

要使浮点数为最大正数，则尾数应该是最大正数 0.1111111（即$1-2^{-7}$）；阶码应该是正数且为最大正数 0111（即 7）。于是，该浮点数格式所能表示的最大正数为

0111	01111111

$2^7\times(1-2^{-7})=2^7-1$

要使浮点数为绝对值最小负数，则尾数应该是绝对值最小负数，且是规格化尾数，故为-0.1000000（即-2^{-1}），其补码表示为 1.1000000；阶码应该是绝对值最大负数 1000（即-2^3）。于是，该浮点数格式所能表示的绝对值最小负数为

1000	11000000

$-(2^{-2^3}\times 2^{-1})=-(2^{-8}\times 2^{-1})=-2^{-9}$

要使浮点数为绝对值最大负数，则尾数应该是绝对值最大负数即 –1，其补码表示为10000000；阶码应该是正数且为最大正数 0111（即 7）。于是，该浮点数格式所能表示的绝对值最大负数为

0111	10000000

$-(2^7\times 1)=-2^7$

该浮点数所能表示的最小数是-2^7，能表示的最大数是2^7-1，故该浮点数能表示的数值范围是$-2^7\sim(2^7-1)$。

④ 若浮点数采用上述①格式，字长 16 位，其中阶码含一位阶符共 4 位，以 2 为底，移码表示；尾数含一位数符共 12 位，补码表示，规格化。求真值$(-2^6\times 0.4375)$的浮点数代码。

$$N=(-2^6\times 0.4375)_{10}=-(0.01110000000)_2\times 2^6$$
$$=-(0.11100000000)_2\times 2^5$$

尾数 M 的原码表示为：1.11100000000

尾数 M 的补码表示为：1.00100000000

阶 E 的真值为：$E_{真}=(5)_{10}=(0101)_2$

阶码 E（移码表示）：$E_{移}=2^3+E_{真}=(1000+0101)_2=(1101)_2$

所示，浮点数代码为$(1101,100100000000)_2=(D900)_{16}$。

⑤ 某浮点数格式如上述①格式所示，字长 32 位，阶码 8 位（含一位阶符），以 2 为底，移码表示；尾数 24 位（含一位数符），补码表示，规格化。若浮点数代码为$(BDB40000)_{16}$，求其真值。

$(BDB40000)_{16}=(10111101,101101000000\text{L}\ 00)_2$

阶码 $E_{移}=(10111101)_2=2^7+E_{真}$，所示 $E_{真}=(111101)_2=(61)_{10}$。

尾数 $M_{补}=(1.011010\text{L}\ 0)_2$，所以 $M_{真}=(-0.10011)_2=(-0.59375)_{10}$。

所以，浮点数真值 $N=-(2^{61}\times 0.59375)$。

5．扩展操作码的概念及举例

当指令字长较短时，可以利用某些类指令中地址部分位数的减少，来增加操作码的位数，进而增加指令种类，这种操作码称为可变长度操作码（或称为扩展操作码）。

（1）若某机指令字长 12 位，每个地址字段 4 位，若要求有 12 条二地址指令，问单地址指令最多可有几条？

对二地址指令，操作码有$12-4\times2=4$位，允许定义 2^4 条二地址指令。现在，只要求设 12 条二地址指令，则单地址指令最多可设置为 x 条，用下式可计算出 x。

$$x=(2^4-12)\times2^4=(16-12)\times16=64\ \text{（条）}$$

注意：这 64 条单地址指令的操作码均是 8 位。

（2）某机指令字长 16 位，有三个地址段 A_1、A_2 和 A_3，每个地址段 4 位。若采用扩展操作码来扩充指令种类，使得指令系统中有 15 条三地址指令，13 条二地址指令，47 条单地址指令。问：

① 若要给该机指令系统增加零地址指令，最多能增加多少条？

零地址指令最多可增加 x 条：

$$\begin{aligned}x&=\left\{[(2^4-15)\times2^4-13]\times2^4-47\right\}\times2^4\\&=\left\{[(16-15)\times16-13]\times16-47\right\}\times16\\&=16\end{aligned}$$

② 各类指令的操作码位数是多少？

三地址指令的操作码为 4 位，二地址指令的操作码为 8 位，单地址指令的操作码为 12 位，零地址指令的操作码为 16 位。

③ 若零地址指令只设 11 条，请拟出各字段分配方案。

以示意图表示扩展操作码各字段代码分配方案（见图3.2.1）。

15 12 11 8 7 4 3 0

OP	A_1	A_2	A_3	指令格式
0000 … 1110	A_1 … A_1	A_2 … A_2	A_3 … A_3	15 条三地址指令
1111 … 1111	0000 … 1100	A_2 … A_2	A_3 … A_3	13 条二地址指令
1111 … 1111 1111 … 1111 1111 … 1111	1101 … 1101 1110 … 1110 1111 … 1111	0000 … 1111 0000 … 1111 0000 … 1110	A_3 … A_3 A_3 … A_3 A_3 … A_3	47 条单地址指令
1111 … 1111	1111 … 1111	1111 … 1111	0000 … 1010	11 条零地址指令

图 3.2.1　扩展操作码示意图

6．指令地址结构与简化方法

（1）指令的地址结构与简化方法？

指令的地址结构是指指令中明显给出了几个地址，哪几个地址。简化地址结构的方法就是采用隐地址，使指令中的显地址个数减少。

注意：在指令的地址部分是没有一个地址段来描述隐地址的，一般由指令代码（如操作码）来隐含约定有效地址。有的教材将这种有效地址（如操作数地址）由隐地址方式给出称为隐含寻址。本书所对应的教材《计算机组成原理（第 2 版）》（ISBN 978-7-121-09546-7）不用这个概念，故寻址方式是指显地址的寻址方式，在指令的地址部分有相应的字段来描述，这些寻址方式得到的有效地址均是显地址。例如，对间接寻址方式而言，认为有效地址是隐藏在寄存器或主存单元中，它就是隐地址，这是错误的。因为采用间接寻址方式，指令中必须有一个地址段来指明它，故间接寻址获得的有效地址仍属显地址。

（2）指令中采用隐地址是简化地址结构的途径，你能举出哪些例子？

从原理方面看，指令中要给出的地址应有源地址 1 和源地址 2，以提供两个操作数或操作数地址；有结果存放地址（称为目的地址）；有后继指令地址，以提供下一条指令的地址。采用隐地址方式可减少指令中的地址个数，有如下几种方法。

① 在 CPU 中设置程序计数器 PC，用于放后继指令地址，该地址在指令中不再出现，故后继指令地址是采用隐地址方式给出的。

② 用二地址指令完成双操作数运算，每个地址存放一个操作数，运算结果放某一操作数地址中，该地址通常称为目的地址。于是，二地址指令中给出一个源地址、一个目的地址，另一源操作数的地址由目的地址隐含约定；这个源地址是采用隐地址方式给出的。

③ 单地址指令完成双操作数运算时，另一个操作数（即目的操作数）由指令隐含约定，如隐含约定在累加器 AC 中，在指令中无地址段描述它，这个地址也是隐地址方式给出的。

④ 零地址指令是没有显地址的，源地址和目的地址均是指令隐含约定的，故零地址指令是采用隐地址方式来表示源地址和目的地址的。

（3）减少指令中地址部分的位数有哪些途径？

指令字长由操作码位数和地址部分的位数构成。地址部分可能有多个地址，减少它的位数有两个途径：减少地址的个数和减少一个地址的位数。

① 对地址部分的几个地址将其部分或全部采用隐地址方式，则可减少地址部分的位数，甚至不需要地址部分（如零地址指令）。

② 采用寄存器寻址、寄存器间址、自增型寄存器间址、自减型寄存器间址等，以寄存器为基础的寻址方式可大大减少一个地址（段）的信息位数，因为指令中表示这些寄存器需要的二进制代码位数较少。

7．变址寻址与基址寻址

（1）变址寻址与基址寻址的主要异同点。

相同点：

① 都是通过相似的地址计算来获得有效地址。

② 当指令代码确定后，两种寻址方式都提供操作数地址的可变性

③ 指令中都设有形式地址。

不同点：

① 有效地址的基准量。变址寻址是以形式地址提供有效地址的基准量，受指令字长限制，其基准地址范围受限，而位移量（或偏移量）由变址寄存器提供，范围较大；基址寻址是以基址寄存器内容作为有效地址的基准量，故基址范围较大，但位移量（或偏移量）由形式地址提供，故位移量范围较小。

② 寻址方式的应用方面。变址寻址立足于面向用户，可用于访问字符串、数组、表格等批量数据（或其中的某些元素）；基址寻址方式立足于面向系统（软件），解决程序重定位或有限字长指令中扩大寻址空间等。

（2）变址寻址方式的应用例子

假设要将主存 200～203 号单元内容复制到 500～503 单元，主存按字编址。计算机即教材第 3 章的模型机。请用模型机指令系统编制实现上述复制功能的程序。

假设通用寄存器 R_2 和 R_3 分别放着主存地址 B 和 A，则可用如下程序来实现将 200～203 号单元内容复到 500～503 单元。

	EOR R_0，R_0	实现 $0 \rightarrow R_0$
A：	MOV $X(R_0)$，$X(R_0)$；	第一次执行时，将 200 单元内容传送到 500 单元 第二次执行时，将 201 单元内容传送到 501 单元 本指令在程序中执行 4 次，复制工作就全部完成
	1NC R_0 ；	R_0 内容加 1，即修改变址量（即位移量），计 A 地址指令执行次数
	MOV R_1 R_0 ；	将 R_0 传到 R_1 保存
	SUB R_0，(PC)+ ；	相减，(PC)+是立即寻址，立即数是 4。若相减后(R_0)=0，则置特征位 Z=1，否则 Z=0
	JMP R_2，Z ；	判别 A 单元指令是否执行了 4 次，是 4 次，Z 为 1，转 B，否则顺序执行
	MOV R_0 R_1 ；	复制工作未全部完成，将 A 地址指令已执行次数送 R_0
	JMP R_3 ；	转 A 继续复制
B：	……；	复制完成，执行下面程序

在程序中，“MOV $X(R_0)$, $X(R_0)$”和“INC R_0”两条指令是复制工作的核心，它们后面的几条指令用于判别复制工作是否全部结束。

在模型机指令系统中，由于指令字长较短，放不下变址寻址的形式地址、立即寻址的立即数和直接寻址的直接地址，因此将它们分别放在该指令的下一个单元。故本程序中“MOV $X(R_0)$, $X(R_0)$”和“SUB R_0, $(PC)_+$”指令在主存中分别占用 3 个和 2 个字单元，示意如下：

A

MOV	R_0 号	$X(R)$	R_0 号	$X(R)$
200（源寻址的形式地址）				
500（目的寻址的形式地址）				

SUB	R_0 号	R	PC 号	(R)+
4（立即数）				

首次执行 A 单元指令时，源寻址的有效地址=200+(R_0)=200+0=200

目的寻址的有效地址=500+(R_0)=500+0=500

于是第一次执行 A 单元指令时便将 200 号单元内容复制到 500 号单元，然后执行“INC R_0”指令，R_0 内容加 1（即 R_0 内容为 1）。当第二次执行“MOV $X(R_0)$, $X(R_0)$”时，就进行 201 到 501 单元的复制工作，以此类推，直到复制全部结束。

上段程序若用商用机的指令系统来编制，程序通常更短。

8．传送类指令一般分为哪三大类？设置和使用传送类指令时应注意哪些问题？

传送类指令一般分为三类：数据传送指令（实现 CPU 内各寄存器间的传送）、访存指令（主要指读/写主存，实质是主存单元和 CPU 的寄存器之间传送）和 I/O 指令（指 CPU 的寄存器和 I/O 接口寄存器之间的传送）。在设置和使用传送类指令时，应注意传送范围、传送单位、传送的数据类型和源地址与目的地址的寻址方式。

9．计算机是如何借助传送类指令来实现CPU对I/O设备的控制及它们之间的信息交换的？

通过将外设接口的有关寄存器与主存单元统一编址，即给 I/O 接口的有关寄存器分配统一的总线地址，访问这些地址单元就是访问相应的外设接口寄存器。于是，可用传送类指令来做如下事务：

① 用传送指令将放于 CPU 某寄存器中的命令字传至某接口的命令字寄存器，后者的输出去控制外设的操作（如启动或关闭外设等）。

② 用传送指令将外设接口的状态字寄存器内容传至 CPU 的某寄存器，供 CPU 判断外设及接口的工作状态。

③ 用传送指令将 CPU 某寄存器内容，传至外设接口的数据缓冲寄存器。

④ 用传送指令将外设接口的数据缓冲寄存器内容传至 CPU 的某寄存器。

CPU 送信息给外设接口的传送指令，起着输出指令的作用；CPU 接收外设接口寄存器信息的传送指令起着输入指令的作用。

10．外围设备与主存统一编址。

外围设备与主存统一编址的含义已在前面和本章练习题的答案中作了介绍。其具体做法是将主存寻址空间划分为两部分：从 0 单元开始的绝大部分存储空间作为用户可用的主存；剩下的一小部分单元称为主存高端（如 A 地址至主存最大地址），留给 I/O 接口的有关寄存器来编址。例如，若主存按字编址，将 A、A+1 和 A+2 地址分配给某设备接口，分别作为该接口中的命令字寄存器、状态字寄存器和数据缓冲寄存器的编程地址。从 A+3 地址开始，分配一些地址给另一设备，直到所有外围设备接口的有关寄存器都分配有各自的编程地址为止。这样，当 CPU 执行传送指令时，若向地址总线送出的地址小于 A，则是访问主存，否则是访问 I/O 接口的相应寄存器。这种外围设备编址方式使得用户程序及数据不能放到 A 开始的主存高端地址单元中，故用户实际可用的主存编程空间减少；另一方面，若主存容量是做满的，高端存储空间闲置不能用，也是一种浪费。

3.2.2 第 2 章习题解答与解题思路

1．简要解释下列名词术语

【答】

位权：在 r 进位制的数中，每个数位的数码所表示的数值等于该数码乘以一个与它所在数位相关的常数，这个常数称为该位的位权，简称权。

基数：在进位制中，各数位允许选用的数码个数，称为该进位制的基数，等于该进位制各数位所允许的最大数码值加 1。

真值：在数的绝对值之前配上正（+，通常可省略）、负（–）符号表示的数称为该数的真值。例如，用十进制数表示的真值 159 和–132，用二进制数表示的真值为 1011、–1011。

机器数：在计算机内部使用的、连同数的符号一起数码化的数称为机器数。

原码：让数码序列的最高位为符号位（0 表示正，1 表示负），其余部分为数（真值）的绝对值，这个数码序列称为该数的原码表示。

补码：机器数的一种表示方法，如果数为正，则正数的补码与原码形式相同；如果数为负，则负数的补码是将负数原码除符号位不变外，其余各位取反，末位再加 1。

定点数：在计算机中，小数点位置固定不变的数叫做定点数。

浮点数：小数点位置不固定，可随需要浮动的数称为浮点数。

规格化浮点数：指浮点数的尾数部分用带符号定点小数表示，R=2 时，尾数的绝对值满足 $\frac{1}{2} \leqslant |M| < 1$（即小数点后第一位不为零）的浮点数称为规格化浮点数。

ASCII 码：美国信息交换标准码的英文全名的简称，与 ISO646、GB—1988 标准兼容，是 128 个常用字符的数码化表示，如字符 A 的 ASCII 码为 1000001B。

机器指令：在计算机领域中，把用 0，1 代码序列表示的指令称为机器指令，是计算机硬件能直接识别和执行的指令。

指令系统：一台计算机所能执行的全部指令称为该机的指令系统或指令集合。

地址结构：指令的地址结构是指在指令中明确给出的地址。

显地址：在指令代码中明显给出的地址，如在指令中写明操作数的主存单元地址或寄存器号，则这种地址称为显地址。

隐地址：在指令中不明显给出地址码，地址以隐含方式约定，这种隐含约定的地址称为隐地址。

寻址方式：指令中以什么方式提供操作数或操作数地址，称为寻址方式。

立即寻址：由指令直接给出操作数，在取出指令的同时也就取出了可以立即使用的操作数，这种寻址方式称为立即寻址。

直接寻址：由指令直接给出操作数地址，根据该地址可以从主存（或寄存器）中取出操作数，或向主存（或寄存器）写入数据，这种寻址方式称为直接寻址。

寄存器寻址：在指令中给出寄存器号，从该寄存器号所指的寄存器中取出操作数或将数据传送到该寄存器号所指的寄存器中。这种寻址方式实为寄存器直接寻址。

间接寻址：在指令中给出间址单元地址码（即操作数地址的存放单元地址），按照该地址访问主存中该间址单元，从中读取操作数地址，接着按操作数地址再次访问主存，从该单元中读取或向该单元写入操作数。

寄存器间址：由指令给出寄存器号，在该寄存器号所指定的寄存器中存放着操作数地址，按此地址访问主存，读取或写入操作数。

间址单元：在间接寻址方式中，存放操作数地址的主存编址单元称为间址单元。

变址寻址：在指令中的地址部分给出一个形式地址，并且指定一个寄存器作为变址寄存

器，将变址寄存器的内容（称为变址量）与形式地址相加，得到操作数地址（称为有效地址）；按有效地址访问主存，从相应的主存单元中读得操作数或向该单元写入数据。

基址寻址：在指令中给出一个形式地址（作为位移量），并且指定一个寄存器作为基址寄存器（该基址寄存器内容作为基准地址）；将基址寄存器内容和形式地址相加，其和作为操作数有效地址；按有效地址访问主存，从该单元读取操作数或向该单元写入数据。

相对寻址：指用程序计数器 PC 的内容作为基准地址，指令中给出的形式地址作为位移量的基址寻址方式。

页面寻址：将程序计数器 PC 的高位段作为操作数有效地址的高位段，指令中给出的形式地址作为操作数有效地址的低位段，将这两部分拼接构成操作数有效地址，这种寻址方式称为页面寻址方式。

堆栈：一种按“后进先出”（或称“先进后出”）存取顺序进行存取的存储结构。

栈顶：堆栈是一个连续的存储区，其一端固定称为栈底，存放最先压入的数；堆栈的另一端是浮动的，称为栈顶，对堆栈的读写都是对栈顶单元进行的；对堆栈的寻址也就是对栈顶单元的寻址，随着堆栈操作的进行，栈顶位置也发生变化。

堆栈指针：指用于指向栈顶位置的寄存器 SP，堆栈指针 SP 的内容是栈顶单元地址。

CISC：具有复杂指令集合的计算机称为 CISC（Complex Instruction Set Computer）。

RISC：采用精简指令系统的计算机称为 RISC（Reduced Instruction Set Computer）。

算术移位：指对具有数值大小的数，将数码位置左、右移动，使其数值发生变化，但数的符号位不变的一类移位操作。

逻辑移位：指将（二进制）代码序列视为纯逻辑意义上的代码组合，只是将数码位置循环移动或非循环移动，使数码位置发生变化，但没有正负性质，也没有数值大小变化的问题。

2．将二进制数$(1111010.00111101)_2$转换为八进制与十六进制数。

【答】

将二进制数转换为八进制数或十六进制数，可分别采用二-八缩写形式或二-十六缩写形式将二进制数分段对应转换即可。

对于二-八缩写形式，则是 3 位二进制数对应一位八进制数，而二-十六缩写形式，则是 4 位二进制数对应一位十六进制的数。分段时，以小数点为基准，向左每 3 位（或 4 位）一组分段，高位不够补 0；向右每 3 位（或 4 位）一组分段，低位不够补 0。

根据上面所述，本题答案如下：

$$(1111010.00111101)_2=(172.172)_8$$

$$(1111010.00111101)_2=(7A.3D)_{16}$$

3．将二进制数$(101010.01)_2$转换为十进制数与BCD码。

【答】

求二进制数中为 1 的各位的权值之和，则得到该二进制数的相应十进制数：

$$(101010.01)_2=(2^5+2^3+2^1+2^{-2})_{10}=(42.25)_{10}$$

将十进制数的各位均用四位二进制代码表示，即得到 BCD 码：

$$(101010.01)_2=(42.25)_{10}=(01000010.00100101)_{BCD}$$

注意：在 BCD 码中，最高位 0 和最低位的 0（包括与它们相邻的 0）不能省掉。本例最高位的 0 不能省掉。

4．将八进制数$(37.2)_8$转换为十进制数与BCD码。

【答】

方法一：由按权相加法计算，即将八进制数展开成多项式求和的形式，求得的和为相应的十进制数，再写为 BCD 码。

$$(37.2)_8=(3\times8^1+7\times8^0+2\times8^{-1})_{10}=(31.25)_{10}$$

$$(37.2)_8=(31.25)_{10}=(00110001.00100101)_{BCD}$$

方法二：将八进制数转换为二进制数（即每位八进制数码用 3 位二进制数表示，或用二—八缩写形式书写），再将二进制数转换为十进制数与 BCD 码。

$$(37.2)_8=(011111.010)_2=(2^4+2^3+2^2+2^1+2^0+2^{-2})_{10}=(31.25)_{10}=(00110001.00100101)_{BCD}$$

5．将十六制数$(AC.E)_{16}$转换为十进制数与BCD码。

【答】

方法一：将十六进制数展开成多项式求和的形式，求得的和即为相应的十进制数，再书写成 BCD 码。

$$(AC.E)_{16}=10\times(16)^1+12\times(16)^0+14\times(16)^{-1}{}_{10}$$

$$=(160+12+0.875)_{10}=(172.875)_{10}$$

$$=(000101110010.100001110101)_{BCD}$$

方法二：将十六进制数转换为二进制数（即每位十六进制数码用 4 位二进制数表示或称为每位用二-十六缩写形式书写），然后把二进制数转换为十进制数，进而可书写为 BCD 码。

$$(AC.E)_{16}=(10101100.1110)_2=(2^7+2^5+2^3+2^2+2^{-1}+2^{-2}+2^{-3})_{10}$$

$$=(128+32+8+4+0.5+0.25+0.125)_{10}$$

$$=(172.875)_{10}$$

$$=(000101110010.100001110101)_{BCD}$$

6．将十进制数$(75.34)_{10}$转换为 8 位二进制数、八进制数及十六进制数。

【答】

方法一：先用减权定位法将$(75.34)_{10}$转换为二进制数，再转换为八进制数和十六进制数。

整数部分减权比较	X_i	位权
75−64=11	1（高位）	64
11＜32	0	32
11＜16	0	16
11−8=3	1	8
3＜4	0	4
3−2=1	1	2
1−1=0	1（低位）	1

整数部分转换结果：$(75)_{10}=(1001011)_2$。

小数部分减权比较	X_i	位权
0.34＜0.5	0	0.5
0.34–0.25=0.9	1	0.25
…	…	…

所以，$(75.34)_{10}=(1001011.01\ldots)_2$。

题目要求转换为 8 位二进制数，故对上述结果采用 0 舍 1 入的舍入法，得到：

$$(75.34)_{10}\approx(1001011.1)_2=(113.4)_8=(4B.8)_{16}$$

方法二：按除基取余法将整数 75 转换为二进制整数；按乘基取整法将$(0.35)_{10}$转换为二进制小数。两部分合起来即得到结果，再转为八进制数和十六进制数。

整数部分转换：

	余数	二进制数位
2 | 75		
2 | 37	1（低位）	$X_0=1$
2 | 18	1	$X_1=1$
2 | 9	0	$X_2=0$
2 | 4	1	$X_3=1$
2 | 2	0	$X_4=0$
2 | 1	0	$X_5=0$
0	1（高位）	$X_6=1$

转换结果为$(75)_{10}=(1001011)_2$。

小数部分转换：

小数部分乘以 2	取整数部分
0.34×2=0.68	0
0.68×2=1.36	1
…	…

所以，$(75.34)_{10}=(1001011.01\ldots)_2$。

题目要求转换为 8 位二进制数，故对上述结果采用 0 舍 1 入的舍入法，得到：

$$(75.34)=(1001011.01\ldots)_2\approx(1001011.1)_2=(113.4)_8=(4B.8)_{16}$$

7．将十进制数 $\frac{13}{128}$ 转换为二进制数。

【答】

将以 2^n 为分母的分数转换为二进制数时，可用简单的方法进行。

$$\left(\frac{13}{128}\right)_{10}=\left(\frac{13}{2^7}\right)_{10}=(13\times 2^{-7})_{10}=(1101\times 0.0000001)_2=(0.0001101)_2$$

8．分别写出下列各二进制数的原码与补码，字长（含 1 位数符）为 8 位。

（1）0　　（2）–0　　（3）0.1010

（4）–0.1010　　（5）1010　　（6）–1010

【答】

题目中给出的数均是真值，其位数都不到 7 位，因此对应的原码和补码均不到 8 位（包括 1 位符号位）。但题目中要求的字长是 8 位，因此应将对应的原码和补码写成 8 位的机器数，如表 3-2-1 所示。

表 3-2-1　真值、原码、补码的对应关系

机器数 / 真值	原　码	补　码
（1）0	整数：00000000 小数：0.0000000	整数：00000000 小数：0.0000000
（2）–0	整数：10000000 小数：1.0000000	
（3）0.1010	0.1010000	0.1010000
（4）–0.1010	1.1010000	1.0110000
（5）1010	0 0001010	00001010
（6）–1010	1 0001010	11110110

注：（3）、（4）题的 0.1010、–0.1010 其原码补码均只有 5 位，不够 8 位，在它们尾数末位后补三个“0”，即成为字长 8 位的机器数。

（5）、（6）题的 1010、–1010，其原码、补码均只有 5 位，不够 8 位，在它们尾数前面（即符号位后）补上三位二进制数码：正数补 000；负数补码补 111，负数原码补 000。

9．若$X_{补}$=0.1010，写出其$X_{原}$与真值X。

【答】

$X_{原}$=0.1010，真值 X=0.1010。

10．若$X_{补}$=1.1010，写出$X_{原}$与真值X。

【答】

$X_{原}$=1.0110，X=–0.0110。

11．某定点小数字长 16 位，含 1 位符号，原码表示，分别写出下列典型值的二进制代码与十进制真值。

（1）非零最小正数　　（2）最大正数

（3）绝对值最小负数　　（4）绝大值最大负数

【答】

四种典型值如表 3-2-2 所示。

表 3-2-2　定点小数原码典型值

	原　　码	二进制真值	十进制真值
（1）非零最小正数	0.000000000000001	0.000000000000001	2^{-15}
（2）最大正数	0.111111111111111	0.111111111111111	$1-2^{-15}$
（3）绝对值最小负数	1.000000000000001	–0.000000000000001	-2^{-15}
（4）绝对值最大负数	1.111111111111111	–0.111111111111111	$-(1-2^{-15})$

12．某定点小数字长 16 位，含 1 位符号，补码表示，分别写出下列典型值的二进制代码与十进制真值。

（1）非零最小正数　　（2）最大正数

（3）绝对值最小负数　　（4）绝大值最大负数

【答】

四种典型值如表 3-2-3 所示。

表 3-2-3　定点小数补码典型值

	补　　码	二进制真值	十进制真值
（1）非零最小正数	0.000000000000001	0.000000000000001	2^{-15}
（2）最大正数	0.111111111111111	0.111111111111111	$1-2^{-15}$
（3）绝对值最小负数	1.111111111111111	–0.000000000000001	-2^{-15}
（4）绝对值最大负数	1.000000000000000	–1	–1

13．某定点整数字长 16 位，含 1 位符号，补码表示，分别写出下列典型值的二进制代码与十进制真值。

（1）非零最小正数　　（2）最大正数

（3）绝对值最小负数　　（4）绝大值最大负数

【答】

四种典型值如表 3-2-4 所示。

表 3-2-4　定点整数补码典型值

典型值	补　　码	二进制真值	十进制真值
（1）非零最小正数	0000000000000001	1	1
（2）最大正数	0111111111111111	111111111111111	$2^{15}-1$
（3）绝对值最小负数	1111111111111111	–1	–1
（4）绝对值最大负数	1000000000000000	–1000000000000000	-2^{15}

14．某浮点数字长 16 位，其中阶码 6 位，含 1 位阶符，补码表示，以 2 为底；尾数 10 位，含 1 位数符，补码表示，规格化。分别写出下列各典型值的二进制代码与十进制真值。

（1）非零最小正数　　（2）最大正数

（3）绝对值最小负数　　（4）绝大值最大负数

【答】

四种典型值用表 3-2-5 表示。

表 3-2-5　浮点数典型值

典型值	浮点数代码	真值
（1）非零最小正数	100000，0.10…0	$(2^{-2^5})(2^{-1})$
（2）最大正数	011111，0.11…1	$(2^{2^5-1})(1-2^{-9})$
（3）绝对值最小负数	10000，1.10…0	$-(2^{-2^5})(2^{-1})$
（4）绝对值最大负数	011111，1.00…0	$(2^{2^5-1})(-1)$

15．若采用图 2-4 的浮点数格式，字长 16 位。其中阶码 6 位，含 1 位阶符，补码表示，以 2 为底；尾数 10 位，含 1 位数符，补码表示，规格化；某浮点数代码为$(A27F)_{16}$，写出其十进制真值。

【答】

浮点十六进制代码：A27F。

浮点二进制代码：101000,1001111111。

阶码（补码）为：101000。

阶码二进制真值：–11000。

阶码十进制真值：–24。

尾数补码：1.001111111。

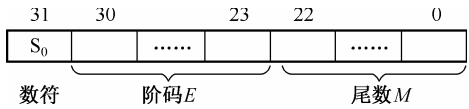
尾数二进制真值：–0.110000001。

尾数十进制真值：$-(2^{-1}+2^{-2}+2^{-9})$。

浮点十进制数真值：$-(2^{-1}+2^{-2}+2^{-9})2^{-24}=-(2^{-25}+2^{-26}+2^{-33})$。

16．若采用如下所示的 IEEE754 短浮点数格式，请将十进制数 37.25 写成浮点数，并且写出其二进制代码序列。

IEEE754 短浮点数格式为：



【答】

将十进制数 37.25 转换为二进制数 100101.01，按 IEEE754 标准的短实数浮点格式要求，将 100101.01 表示为 1.0010101×2^5，故浮点数阶码的真值 $e=5$。于是，按 IEEE754 标准，得到：

数符 S_0=0

阶码（移码表示）$E=(e+127)_{10}=(5+127)_{10}=(132)_{10}=(10000101)_2$

M=001010100000…00。

最后得到 32 位浮点数的二进制数代码序列为：

01000010100101010000000000000000

17．简化地址结构的基本途径是什么？

【答】

在指令中减少显地址的数量，即使用隐地址方式给出地址，指令中的地址（段）个数就可减少。

18．减少指令中一个地址信息的位数的方法是什么？

【答】

采用寄存器寻址、寄存器间址等，以寄存器为基础的寻址方式可以大大减少指令中一个地址的信息位数。

19．某主存储器部分单元的地址码与存储器内容对应关系如下：

地址码	存储内容
1000H	A307H
1001H	0B3FH
1002H	1200H
1003H	F03CH
1004H	D024H

（1）若采用寄存器间址方式读取操作数，指定寄存器 R_0 的内容为 1002H，则操作数是多少？

（2）若采用自增型寄存器间址方式(R_0)+读取操作数，R_0 内容为 1000H，则操作数是多少？指令执行后 R_0 的内容是多少？

（3）若采用自减型寄存器间址方式–(R_1)读取操作数，R_1 内容为 1003H，则操作数是多少？指令执行后 R_1 的内容是多少？

（4）若采用变址寻址方式 $X(R_2)$读取操作数，指令中给出形式地址 d=3H，变址寄存器 R_2 内容为 1000H，则操作数是多少？

【答】

（1）操作数是 1200H。

（2）操作数是 A307H，指令执行后 R_0 的内容变为 1001H。

（3）操作数是 1200H，指令执行后 R_1 的内容为 1002H。

（4）操作数为 F03CH。

20．对 I/O 设备的编址方法有哪几种？请简要解释。

【答】

对 I/O 设备的编址方法实际上就是对 I/O 接口中有关寄存器及相应部件的编址方法，有两大类型。

（1）为外围设备单独编址。

早期是为每台 I/O 设备分别分配一个设备码，每个设备下属多个接口寄存器，在 I/O 指令中给出设备码，并指明接口的哪个寄存器，从而实现 CPU 对外设的访问。现在普遍采用为

各 I/O 接口的每个有关寄存器分别分配一种 I/O 端口地址，指令中给出端口地址，也就知道 CPU 访问哪一台设备及其接口寄存器。

（2）外围设备与主存统一编址。

即将各 I/O 接口中的有关寄存器与主存的各编址单元统一编址，为它们分配统一的总线地址。在传送指令中给出这类总线地址，CPU 就可以访问相应的 I/O 设备及其接口寄存器。

21．I/O 指令的设置方法有哪几种？请简要解释。

【答】

通常有三类常见的 I/O 指令设置方法。

（1）在指令系统中设置专门的 I/O 指令，可对外围设备单独分配设备码，或给 I/O 接口的有关寄存器分配专门的端口地址，这种 I/O 指令称为显式 I/O 指令。

（2）采用通用的数据传送指令实现 I/O 操作，相应地将外围设备接口的有关寄存器与主存统一编址。这种 I/O 指令是隐含在传送指令中，所以又称为隐式 I/O 指令。

（3）通过 I/O 处理器（或 I/O 处理机）控制 I/O 操作。这种方式下，I/O 指令可分为两级：CPU 调用 IOP 的指令和 IOP 本身的指令。

3.3 第 3 章习题解答及解题思路

3.3.1 第 3 章重点和难点解析

本章的重点和难点主要包括：原码乘法和补码乘法的运算规则，原码除法和补码除法的运算规则，浮点加减运算流程（对阶、规格化），组合逻辑控制方式和微程序控制方式的基本思想，同步控制方式和异步控制方式的基本概念，模型机组合逻辑控制器的时序划分，模型机各类指令流程和微命令序列，模型机微程序的编写方法等。下面通过几个例子，对这些重点和难点问题进行分析、说明。

1．原码乘法和补码乘法的运算规则

乘法运算规则主要说明了操作数和结果的形式、对符号位的处理、判断位的设置、操作内容、操作步数等。可以将原码乘法和补码乘法按这几项作对比（见表 3-3-1），便于理解和掌握。

在补码乘法中，当乘数的符号位和尾数最高位是 0.0 或 1.1 时，只作 n 步操作，不对结果进行修正。当乘数的符号位和尾数最高位是 0.1 或 1.0 时，需要增加第 n+1 步对结果进行修正，前者作+B 修正，后者作–B 修正。注意：修正后不再移位。

2．原码除法和补码除法的运算规则

与乘法一样，除法运算规则也说明了操作数和结果的形式、对符号位的处理，还说明了如何上商、下一步操作、操作步数、恢复余数、商的校正等问题（见表 3-3-2）。

表 3-3-1　原码乘法和补码乘法的运算规则

原码一位乘法	补码一位乘法
操作数和结果用原码表示； 绝对值参加运算，符号单独处理； 设置一位判断位 C_n； 根据 C_n 的状态决定下一步操作： C_n=1　1/2（A+B） C_n=0　1/2（A+0） 作 n 步操作（n 为乘数位数），不对结果修正	操作数和结果用补码表示； 符号参加运算； 设置两位判断位 C_nC_{n+1}； 根据 C_nC_{n+1} 的状态决定下一步操作： C_nC_{n+1} = 00　1/2（A+0） 01　1/2（A+B） 10　1/2（A−B） 11　1/2（A+0） 作 n 步或 n+1 步操作（n 为乘数位数） 第 n+1 步对结果修正

表 3-3-2　原码除法和补码除法的运算规则

原码不恢复余数除法	补码不恢复余数除法
操作数和结果用原码表示； 绝对值参加运算，符号单独处理； 根据余数正负决定商值及下步操作： 余数为正，商 1，下一步作 2A−B 余数为负，商 0，下一步作 2A+B 求 n 位商，作 n 步操作； 若第 n 步余数为负，则恢复余数	操作数和结果用补码表示； 符号参加运算； 根据余数和除数的符号决定商值及下步操作： 余数和除数同号，商 1，下一步作 2A−B 余数和除数异号，商 0，下一步作 2A+B 求 n−1 位商，作 n 步操作； 不恢复余数； 对商校正（商符变反，第 n 位商恒置 1）

注意：在原码不恢复余数除法的运算过程中，若出现余数为负的情况，则不恢复余数。只有当最后一步余数为负时，才作恢复余数的操作，以保证运算结束时得到的余数是绝对值形式。之后，再确定余数的实际符号，即与被除数同号。

在补码不恢复余数除法中，最后一步余数为正为负都不再恢复余数，因为操作数的符号参加运算，最后得到的符号就是结果的正确符号。此外，由于商的末位恒置 1，可能产生误差，但通过余数调整，使结果是正确的，即满足下式

$$\text{被除数 } X = \text{商 } Q \times \text{除数 } Y + \text{余数 } R$$

可用该式对运算结果进行验证。例如，0.10110 ÷ (–0.11111)，分别用原码除法和补码除法得到的结果如下：

原码除法：$Q = -0.10110$　　$R = 0.10110\times2^{-5}$　　$X = Q\times Y + R$

补码除法：$Q = -0.10111$　　$R = -0.01001\times2^{-5}$　　$X = Q\times Y + R$

3．浮点加减运算流程

浮点加减运算流程包括判操作数是否为 0、对阶、尾数加减、结果规格化等阶段，其中，对阶和结果规格化是重要的基本概念。

（1）对阶

需要掌握的内容有：对阶的定义、原因、原则、操作等。

① 什么是对阶？

对阶是指让两个浮点数的阶码相同。

② 为什么要对阶？

对阶用在浮点数的加减运算中。两个浮点数相加减，是将它们的尾数相加减，但必须首先保证其小数点的位置对齐，即它们对应的数位的权值要相同。例如，两个浮点数 $2^2\times0.1001$ 和 $2^3\times0.1101$，它们的小数点的实际位置是：

10.01
110.1

也就是说，两个浮点数的阶码不同，其尾数的小数点的实际位置是没有对齐的，尾数对应位的权值不同，因而不能直接加减。对齐小数点的位置，使它们成为：

010.01
110.1

写成浮点数形式，即 $2^3\times0.0101$ 和 $2^3\times0.1101$。这样，两数阶码相同，尾数便可以直接加减了。

③ 如何对阶？

对阶是将大阶码减小到与小阶码相同？还是将小阶码增大到与大阶码相同？或者任意方式均可？这是一个原则问题。阶码增大或减小，相应地，尾数应右移或左移，才能使浮点数的值不变。由于小阶增大，尾数右移，舍去的是尾数的低位，误差很小；而大阶减小，尾数左移，舍去的是尾数的高位，误差太大，因而是不允许的。所以，对阶的原则是：小阶向大阶对齐。对阶操作是：将小阶加 1，尾数右移 1 位，直到小阶与大阶相等。

（2）结果规格化

需要掌握的内容有：规格化的定义、左规和右规的判断、左规和右规操作等。

① 什么是规格化？

规格化是针对浮点数尾数而言。浮点数尾数的绝对值大于或等于 1/2 而小于 1，这样的浮点数称为规格化浮点数，从代码形式来看，即尾数最高位的绝对值为 1。对于正数补码（包括正、负数原码），尾数最高位为 1；对于负数补码，尾数最高位为 0。

② 在什么情况下进行左规或右规？

当尾数的绝对值小于 1/2，即尾数的符号位和最高位同为 0.0 或 1.1 时，进行左移规格化，将尾数左移 1 位，阶码减 1，直到尾数的绝对值大于或等于 1/2。当尾数的绝对值大于 1，即尾数的两个符号位不同（01 或 10）时，进行右移规格化，将尾数右移 1 位，第二符号位中存放的进位移至尾数最高位，使尾数的绝对值小于 1，并且阶码加 1。

4．组合逻辑控制方式和微程序控制方式

组合逻辑控制方式和微程序控制方式是产生微命令的两种不同方式，前者直接通过硬连逻辑电路产生微命令，后者则通过执行微指令产生微命令。对于这两种控制方式，需掌握它们的基本思想、优缺点及应用场合。

（1）组合逻辑控制方式

① 基本思想

综合化简产生微命令的条件，形成逻辑式，用组合逻辑电路实现；执行指令时，由组合逻辑电路（微命令发生器）在相应时间发出所需微命令，控制有关操作。

上面这段话是对组合逻辑控制方式基本思想的概括，前一部分说明如何形成控制逻辑，后一部分说明如何产生微命令。由于组合逻辑电路反映了产生微命令的条件与微命令之间的

逻辑关系，因而只要条件（逻辑条件如指令信息、状态信息和时间条件）满足，在组合逻辑电路的输出端就能获得所需微命令，用来控制相应操作。

② 优缺点

主要优点：产生微命令的速度较快。主要缺点：其一，设计不规整，设计效率较低，这是因为不同的微命令可能需要不同的条件，使得产生微命令的逻辑电路零乱、不规整，难于实现设计自动化；其二，不易修改、扩展指令系统功能，因为逻辑电路固定连接，修改起来非常麻烦。机器一旦制作出来，硬件功能就固定了，很难进一步扩充功能。

③ 应用场合

用于高速计算机，或小规模计算机。

（2）微程序控制方式

① 基本思想

微程序控制方式是针对组合逻辑控制方式的主要缺点提出来的，它将程序技术和存储逻辑引入 CPU 的构成级，较好地解决了设计的规整性和功能的可修改性问题。其基本思想可归结为以下几点：

- 将一步操作所需的若干微命令编写在一条微指令中，控制实现一步操作。
- 用若干微指令组成一段微程序，解释执行一条机器指令。
- 将微程序事先存放在控制存储器中，执行机器指令时再取出。

② 优缺点

主要优点：设计规整，设计效率高；易于修改、扩展指令系统功能；结构规整、简洁，可靠性高；性价比高。主要缺点：其一，速度较慢，因访存频繁且转移较多；其二，执行效率不高，因未充分发挥数据通路本身所具有的并行操作能力。

③ 应用场合

用于速度要求不高、功能较复杂的机器中，特别适用于系列机。

5．同步控制方式和异步控制方式

注意区分这两种控制方式在定义、特点和应用场合等方面的不同之处。

3.3.2 第 3 章习题解答与解题思路

1．简要解释下列名词术语。

【答】

CPU：由运算器和控制器组成的计算机硬件系统的核心部件，即中央处理器。

运算器：在计算机中，用来对数据进行加工处理的部件。传统运算器包含输入逻辑、算术逻辑运算部件（ALU）、输出逻辑和一部分寄存器。

控制器：在计算机中，用来产生各种控制命令（微命令）、控制全机操作的部件。传统控制器包含微命令产生部件、时序系统和一部分寄存器。

通用寄存器：可由 CPU 编程访问，能实现多种功能的寄存器。例如，可提供操作数，存放运算结果，用作地址指针，作为变址寄存器、基址寄存器、计数器等。

暂存器：为避免破坏通用寄存器的内容，用来暂时存放某些中间结果的寄存器。

指令寄存器 IR：用来存放现行指令的寄存器。当需要执行某条指令时，先将该指令从存储器取出，并存入 IR 中，再对 IR 的内容进行译码。

程序计数器 PC：用来指示指令在存储器中存放位置的寄存器。PC 的内容是指令所在存储单元的地址，取指后，PC 内容增量计数，指向下一条指令的地址。

程序状态字 PSW：用来记录现行程序的运行状态和指示程序工作方式的寄存器。

时序系统：用来产生时序信号（如周期、节拍、脉冲等）的部件，称为时序系统或时序发生器，由一个振荡器和一组计数分频器组成。

微命令：在计算机中，用来控制微操作（如逻辑门的开或关、寄存器的打入或清除等操作）的控制命令，也称为微操作控制信号。

组合逻辑控制：简单地讲，由硬连逻辑电路产生微命令的方式称为组合逻辑控制方式。它的基本思想如下：综合、化简产生微命令的条件，形成相应逻辑式，并用组合逻辑电路实现；执行指令时，由组合逻辑电路（微命令发生器）在相应时间发出所需微命令，控制有关操作。

微程序控制：简单地讲，由微指令译码产生微命令的方式称为微程序控制方式。其基本思想如下：将若干微命令编制成一条微指令，控制实现一步操作；将若干微指令组成一段微程序，解释执行一条机器指令；将微程序事先存放在控制存储器中，执行机器指令时再取出。

数据通路结构：将有关部件连接起来，为这些部件之间的信息传送提供通路的硬件结构。CPU 内部或微型机内部的数据通路结构多采用总线结构。

同步控制：用统一发出的时序信号（周期、节拍、脉冲）控制各项操作的方式，称为同步控制方式。其特点是：有明显时序时间划分，时钟周期时间固定，各步操作的衔接、各部件之间的数据传送受严格同步定时控制。

异步控制：指各项操作根据实际需要安排不同的时间，不受统一时序信号约束的一种控制方式。其特点是：无固定时钟周期划分，各操作间的衔接和各部件之间的信息交换采用异步应答方式。

主设备：申请并掌握总线控制权的设备。

从设备：响应主设备请求，并与之通信的设备。

全加器：含有三个输入量（两个操作数、一个来自低位的进位信号）的二进制加法单元。

并行加法器：用多位全加器实现多位数同时相加的加法器。

进位链：提供进位信号传递通路的硬连逻辑电路。

串行进位：进位信号逐级产生，低位进位向高位传递。

并行进位：各个进位信号同时产生，高位进位不依赖于低位进位。

多重分组跳跃进位：也称为分级同时进位，即将多位全加器分成若干组，组内采用并行进位，组间也采用并行进位。

对阶：让两个浮点数的阶码相等，称为对阶。对阶操作是将小阶增大，尾数右移。

左规：将浮点数的尾数左移，使其成为规格化尾数，称为左规。左规操作是将尾数左移，阶码减小。

右规：将浮点数的尾数右移，使其成为规格化尾数，称为右规。右规操作是将尾数右移，阶码增大。

指令周期：一条指令从取指到执行完所用的全部时间。

工作周期：一个指令周期中，完成某一阶段操作所需的时间，如取指周期、源周期、目的周期、执行周期等。

时钟周期：CPU 执行一步操作所需的时间。时钟周期作为时序基准，在一个计算机中其长度是固定不变的。

微指令周期：读取并执行一条微指令所用的时间。

总线周期：通过总线传送一次数据所用的时间。在同步方式下，一个总线周期可能包含若干个时钟周期。

主存读/写周期：指主存进行连续读/写所允许的最小时间间隔，即两次读/写操作之间的最小间隔。

微指令：将一步操作所需的微命令编写在一串代码中，这串代码称为微指令。微指令由微命令字段和微地址字段组成。

微程序：由若干条微指令组成一段微程序，用来解释执行一条机器指令。

控制存储器：用来存放各微程序段的专用存储器，属于 CPU 范畴而不属于主存范畴。

不译法：也称为直接控制法，即微命令按位给出，不需要对微命令编码和译码。

分段直接编译法：也称为单重定义或显式编码法，即将微指令划分为若干字段，微命令由字段编码直接给出。

分段间接编译法：也称为多重定义或隐式编码法，即将微指令划分为若干字段，微命令由本字段编码和其他字段解释共同给出。

功能转移：将机器指令代码转换成相应的微程序入口地址，称为功能转移。

增量方式：这是一种产生微地址的方式，即以顺序执行为主，后续微地址在现行微地址的基础上增量产生，并配合多种常规转移方式。

断定方式：这也是一种产生微地址的方式，通过直接给定和测试断定相结合来产生后续微地址，即后续微地址的一部分由现行微指令给定，另一部分则由测试判断来确定。

2．用变形补码计算$X_{补}+Y_{补}$，并指出是否有溢出。

（1）$X_{补}$=00.110011　　$Y_{补}$=00.101101

（2）$X_{补}$=00.010110　　$Y_{补}$=00.100101

（3）$X_{补}$=11.110011　　$Y_{补}$=11.101101

（4）$X_{补}$=11.001101　　$Y_{补}$=11.010011

【答】

做补码加法时，将两个操作数直接相加。对于变形补码而言，溢出=第一符号位 ⊕ 第二符号位，因此可以根据运算结果的两个符号位相同或相异，判断是否发生了溢出以及溢出的类型。

（1）

$$\begin{array}{r} 00.110011 \\ +\ 00.101101 \\ \hline 01.100000 \end{array}$$

结果的两个符号位相异，表明发生了溢出；又由于第一符号位为 0 表示结果为正，所以发生了正溢出

（2）
```
  00.010110
+ 00.100101
----------
  00.111011
```
结果的两个符号位相同，为 00，表明未发生溢出，结果为正

（3）
```
  11.110011
+ 11.101101
----------
  11.100000
```
结果的两个符号位相同，为 11，表明未发生溢出，结果为负

（4）
```
  11.001101
+ 11.010011
----------
  10.100000
```
结果的两个符号位相异，表明发生了溢出；又由于第一符号位为 1 表示结果为负，所以发生了负溢出

3．用变形补码计算$X_{补}-Y_{补}$，并指出是否有溢出。

（1）$X_{补}$=00.100011　　$Y_{补}$=00.101101

（2）$X_{补}$=00.110110　　$Y_{补}$=11.010011

（3）$X_{补}$=11.100011　　$Y_{补}$=00.110100

（4）$X_{补}$=11.101101　　$Y_{补}$=11.010011

【答】

做补码减法时，将减数 $Y_{补}$变补，再与被减数 $X_{补}$相加。注意，变补是将 $Y_{补}$的尾数连同符号位一起变补。

（1）将 $Y_{补}$变补，即$-Y_{补}$=11.010011。
```
  00.100011
+ 11.010011
----------
  11.110110
```
结果的两个符号位相同，为 11，表明未发生溢出，结果为负

（2）$-Y_{补}$=00.101101
```
  00.110110
+ 00.101101
----------
  01.100011
```
结果的两个符号位相异，表明发生了溢出；由于第一符号位为 0 表示结果为正，所以发生了正溢出

（3）$-Y_{补}$=11.001100
```
  11.100011
+ 11.001100
----------
  10.101111
```
结果的两个符号位相异，表明发生了溢出；由于第一符号位为 1 表示结果为负，所以发生了负溢出

（4）$-Y_{补}$=00.101101
```
  11.101101
+ 11.101101
----------
  00.011010
```
结果的两个符号位相同，为 00，表明未发生溢出，结果为正

4．用 74181 和 74182 芯片构成一个 64 位ALU，采用分级分组并行进位链结构。画出逻辑图，并注明输入、输出信号。

【答】

首先，计算 74181 芯片的数量。由于 1 片 74181 实现 4 位数运算，所以 64 位数需要 16

片 74181。

其次，计算 74182 芯片的数量。1 片 74182 只能接受 4 个小组的进位辅助函数，并同时输出 4 个组间进位信号，因此将 16 片 74181 分为 4 组，每组 4 片，组内共用 1 片 74182，同时产生组内各 74181 的初始进位。4 片 74182 再共用另 1 片 74182，同时产生 4 个组间进位信号。所以，一共需要 5 片 74182 芯片。

ALU 逻辑图如图3.3.1所示。

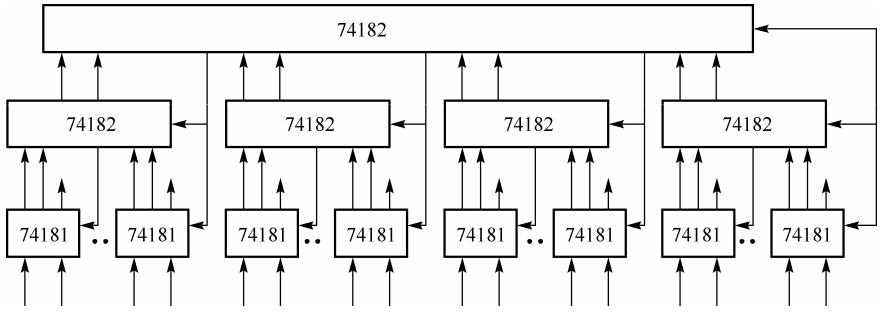


图 3.3.1　ALU 逻辑图

5．用流程图描述下列算法流程。

（1）补码一位乘法

（2）原码两位乘法

（3）原码加减交替除法

（4）补码加减交替除法

（5）浮点加减运算

（6）浮点乘法运算

（7）浮点除法运算

【答】

可参照原码一位乘法的算法流程进行描述。

（1）补码一位乘法

首先，进行初始化，并在乘数末位后设置附加位；然后，根据两位判断位比较的结果进行操作；每做一步操作，计数器加 1，作完 n 步操作后若需修正结果，则再增加一步。流程图如图3.3.2所示。

（2）原码两位乘法

首先，进行初始化，并设置欠账触发器 C_J，与两位乘数一起组成三位判断位；然后，根据三位判断位的状态进行操作；每作一步操作计数器加 1，作完 $n/2$ 步操作后若仍欠账，则再增加一步还账。流程图如图3.3.3所示。

（3）原码加减交替除法

首先，进行初始化，然后将初始余数左移一位减除数，根据余数的正负确定商值，并根据商值决定下一步操作。每做一步操作，计数器加 1，若求 n 位商，则第 n 步操作后如果余数为负，那么再增加一步恢复余数。流程图如图3.3.4所示。

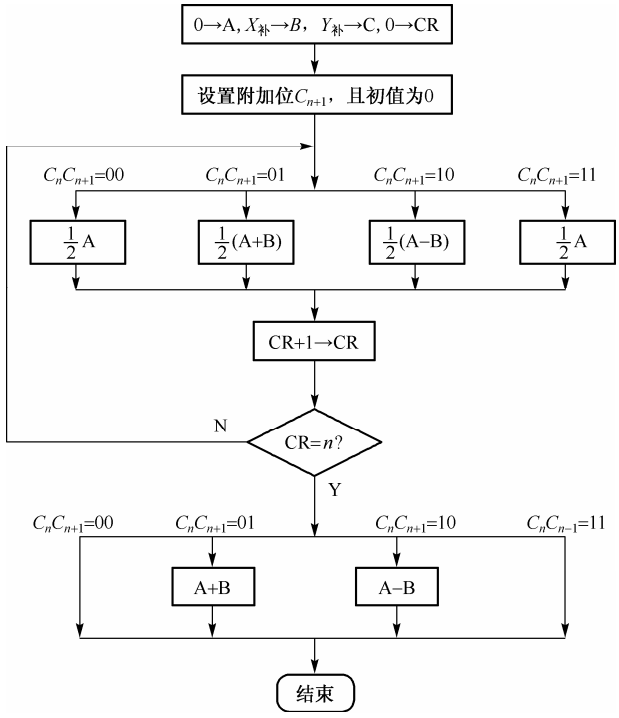


图 3.3.2　补码一位乘法流程

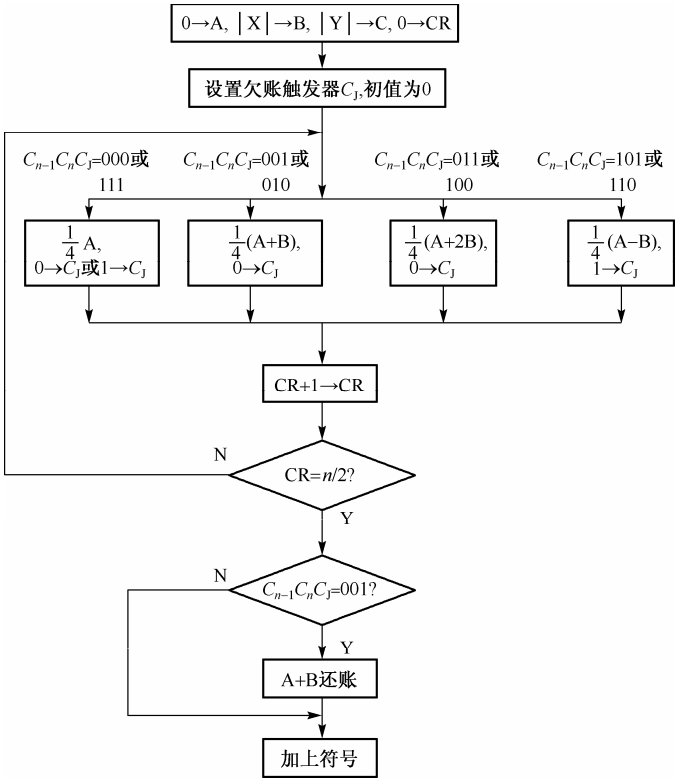


图 3.3.3　原码两位乘法流程

（4）补码加减交替除法

先进行初始化，然后根据初始余数和除数的符号求商符，并根据商符决定下一步操作。以后每步操作都求商值，并且每做一步操作，计数器加 1。当做完第 n-1 步操作后求得 n-1 位假商，再做一步操作求得第步余数。最后对假商进行修正。流程图如图 3.3.5 所示。

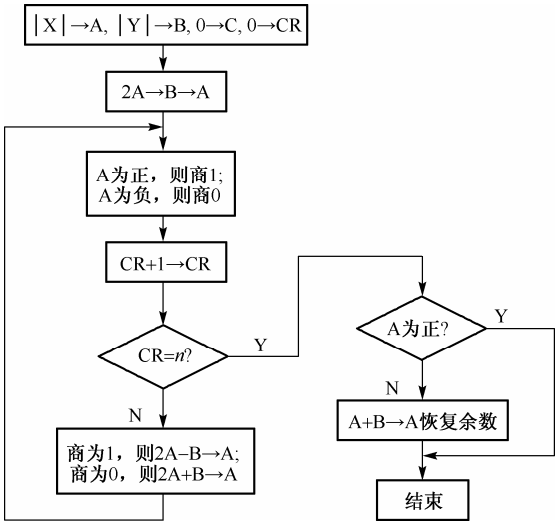


图 3.3.4　原码加减交替除法流程

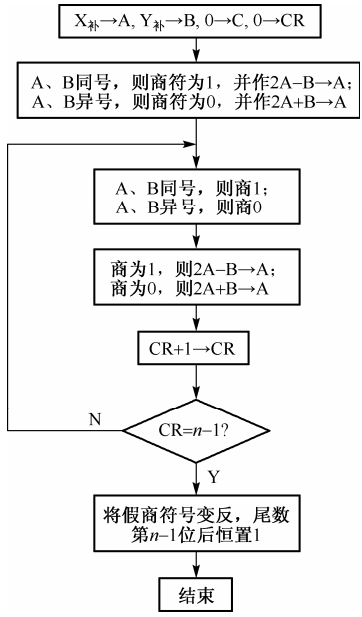


图 3.3.5　补码加减交替除法流程

（5）浮点加减运算

浮点加减运算中，主要的操作是对阶和结果规格化。流程图如图3.3.6所示。

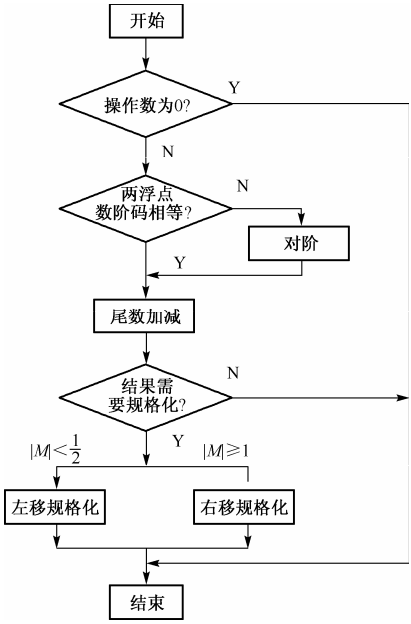


图 3.3.6　浮点加减流程

（6）浮点乘法运算

流程图如图3.3.7所示。

（7）浮点除法运算

流程图如图3.3.8所示。

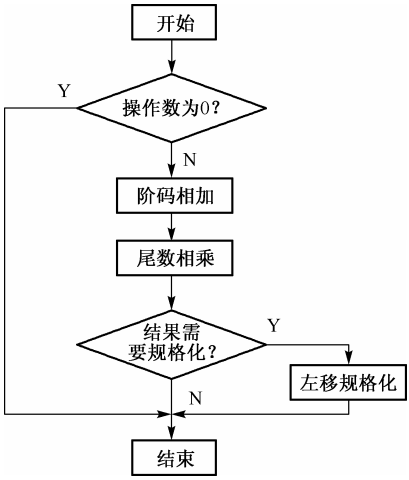


图 3.3.7　浮点乘法流程

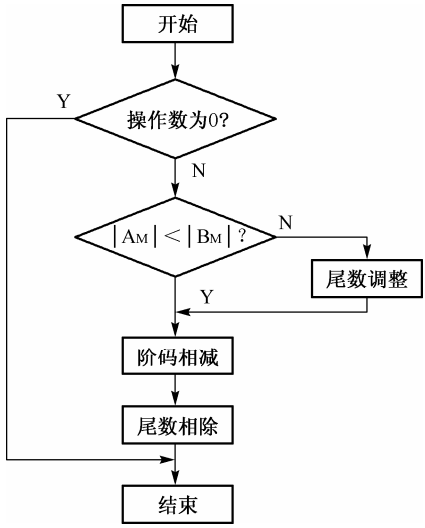


图 3.3.8　浮点除法流程图

操作数为 0 有两种情况：如果被除数为 0，则结果为 0；如果除数为 0，则另行处理。

6．参照图 3.3.9 形式，设计下列乘法器和除法器。

（1）补码一位乘的乘法器。

（2）原码两位乘的乘法器。

（3）原码加减交替除法器。

（4）补码加减交替除法器。

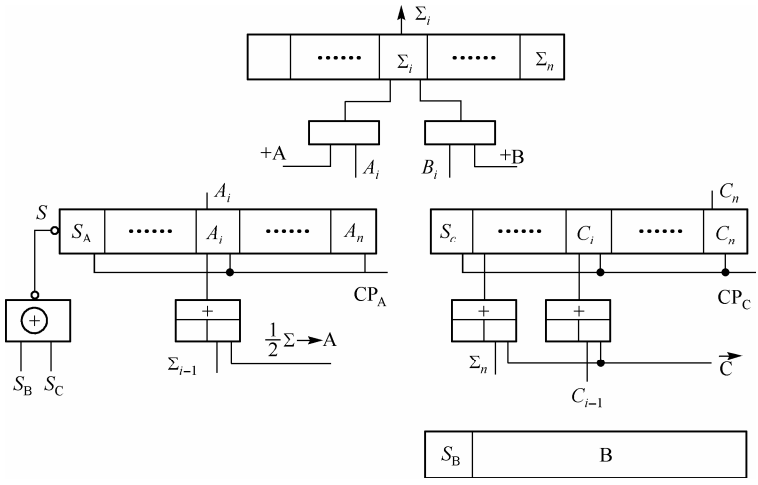


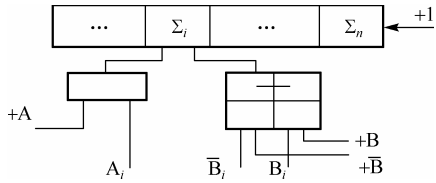
图 3.3.9　原码一位乘法器粗框

【答】

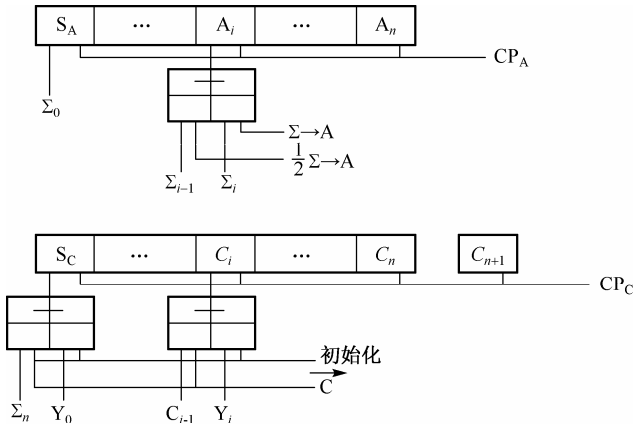
如图3.3.9 所示的原码一位乘法器是在加法器的基础上，加入其输入逻辑和输出逻辑而构成的，因此要设计其他乘法器、除法器，关键在于根据算法所需的控制命令设计相应的输入逻辑和输出逻辑。

（1）补码一位乘的乘法器

加法器的输入逻辑是 A、B 两个输入门，加法器输入端所需的控制命令有：+A、+B、+$\overline{B}$ 、+1。由于 A 端只有一个命令+A，故采用与门；B 端有两个命令+B、+$\overline{B}$，采用与或门。命令+1 加在加法器末端，控制末位加 1。

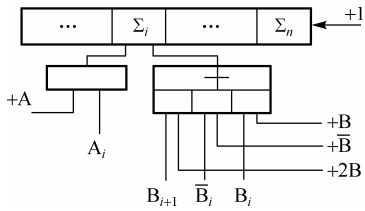


加法器的输出逻辑即是寄存器 A、C 的输入控制门，其所需的控制命令有：Σ→A、Σ/2→A、$\overline{C}$ 、CP_A、CP_C。寄存器 A 的输入控制门采用与或门，实现将累加和右斜1位送 A 寄存器以及将修正结果送 A 寄存器。寄存器 C 的输入控制门也采用与或门，在初始化时将 $Y_{补}$ 送 C 寄存器，在运算过程中实现 C 寄存器的右移。

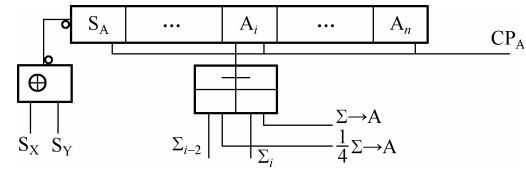


（2）原码两位乘的乘法器

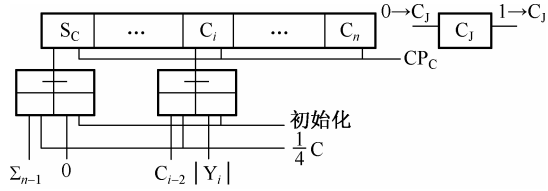
加法器 A 输入端所需的控制命令有+A；B 输入端所需的控制命令有：+B，+2B，+$\overline{B}$，+1。



寄存器 A、C 输入端所需的控制命令有：Σ→A，$\frac{1}{4}$Σ→A，$\frac{1}{4}$C→C，CP_A、CP_C；欠账触发器 C_J 具有置位、复位功能。



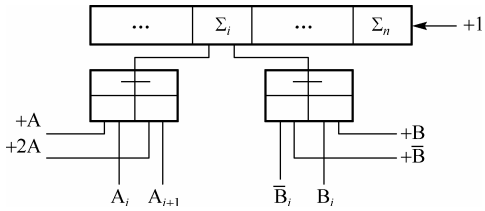
寄存器 A 用来存放乘积的高位部分，其符号位在运算结束后，根据被乘数和乘数的符号进行设置，即同号相乘结果为正、异号相乘结果为负。



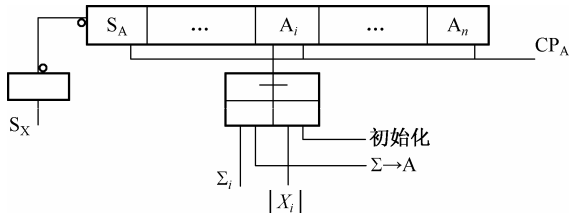
寄存器 C 在初始化时用来存放乘数的绝对值，其符号位设置为 0，以后参加移位，以决定最后是否还账。

（3）原码加减交替除法器

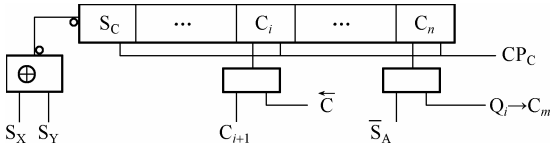
加法器 A 输入端所需的控制命令有：+A、+2A；加法器 B 输入端所需的控制命令有：+B、 $+\overline{B}$ 、+1。



寄存器 A、C 输入端所需的控制命令有：$\Sigma \rightarrow A$、$\overset{\curvearrowright}{C}$、$Q_i \rightarrow C_n$、$CP_A$、$CP_C$。



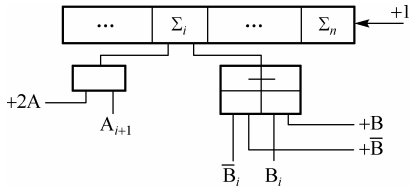
寄存器A用来存放余数，运算结束后，其符号位（表明实际余数的符号）由被除数的符号进行设置，即实际余数与被除数同号。



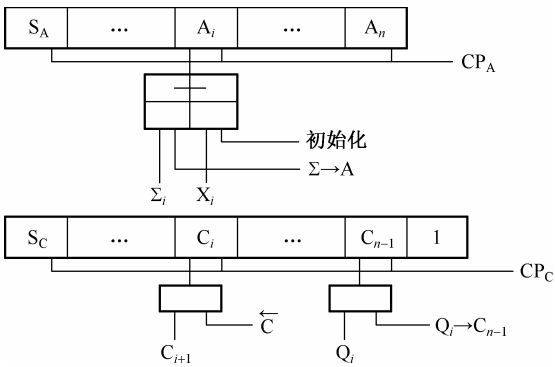
寄存器 C 用来存放商，商值与余数的符号相反，即余数为正商 1，余数为负商 0。每次均在 C_n 位上商，随着 C 左移，将各位商值依次移至 C 中相应位置。商的实际符号由被除数和除数的符号决定，同号相除为正，异号相除为负。

（4）补码加减交替除法器

加法器 A 输入端所需的控制命令有：+2A；加法器 B 输入端所需的控制命令有：+B，$+\overline{B}$，+1。



寄存器 A、C 输入端所需的控制命令有：$\Sigma \rightarrow A$、$\overset{\curvearrowright}{C}$、$Q_i \rightarrow C_{n-1}$、$CP_A$、$CP_C$。



每次均在 C_{n-1} 位上商，商的末位则恒置 1。商值 Q_i 由余数和除数的符号决定，同号商 1，异号商 0。运算结束后，通过 S_C 的置 1、置 0 端将商符变反，实现对商符的校正。

7．某乘法器基本字长 8 位（含 1 位数符），运算时可扩展为双符号位。请参照图 3-31 结构，画出乘法器的完整逻辑图。

【答】

假设该乘法器用来实现原码一位乘法。

加法器设置 8 位，其输入端控制命令有+A、+B。

寄存器 A 存放累加和，设置 8 位，含 1 位符号位 S_A。由于 S_A 可能被累加时产生的进位所占用，因而另设置一位触发器 A_0，其值为 0，右移时移入 S_A，以保证 A 始终以绝对值形式参加运算。A 输入端的控制命令有 $\frac{1}{2}\Sigma \rightarrow A$、$CP_A$。

寄存器 B 存放被乘数的绝对值，设置 8 位，含 1 位符号位 S_B。B 输入端的控制命令为 CP_B。

寄存器 C 存放乘数的绝对值，设置 8 位，含 1 位符号位 S_C。在运算过程中没有使用 S_C。C 输入端的控制命令有 $\overset{\curvearrowleft}{C}$、$CP_C$。

该乘法器的逻辑图如图 3.3.10 所示。

8．试比较同步控制和异步控制的特点及应用场合。

【答】

同步控制和异步控制是用来对操作时间进行控制的两种方式。

同步控制的特点主要表现在三方面。第一，有明显的时序时间的划分，如周期、节拍、

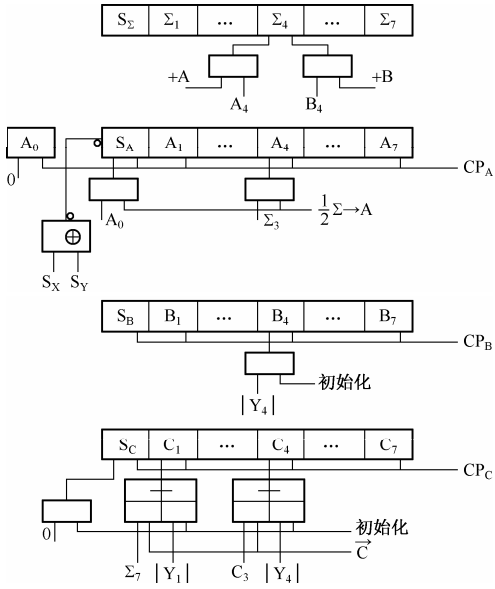


图 3.3.10　乘法器的逻辑图

脉冲等。第二，时钟周期（节拍）时间固定，这是同步控制方式最主要的特点，因为同步控制采用统一时序，时钟周期长度作为控制一步操作所需的基本时间单位，在系统中一经确定，便不能随意改动。第三，各操作之间的衔接、各部件之间的数据传送采用严格的同步定时控制，即时间一到，自动从上一操作转到下一操作，或在规定的时间内完成一次数据传送。同步方式主要应用在 CPU 内部、设备内部或系统总线传送等场合，通常要求挂接在总线上的设备速度差异不大、传送时间确定、传送距离较短等。

异步控制的特点主要表现在两方面。第一，无固定时钟周期的划分，这是异步控制方式最主要的特点，操作所需的时间根据实际需要安排，可长可短。第二，各操作之间的衔接、各部件之间的数据传送采用异步应答方式，一方发出请求或询问信号，另一方收到信号后才开始新的操作，这样可以合理安排操作时间。由于异步控制较为复杂，所以常应用于异步总线的传送操作，挂接在总线上的设备速度差异可以较大，传送时间可以不确定，传送距离可以较远。

9．试比较组合逻辑控制和微程序控制的优缺点及应用场合。

【答】

组合逻辑控制和微程序控制是用来控制如何产生微命令的两种方式。

组合逻辑控制方式是直接通过逻辑门电路产生微命令的，因而它的主要优点是产生微命令的速度很快。其主要缺点有两点：其一是设计不规整，因而难于实现设计自动化；其二是采用硬连逻辑，不易修改和扩展指令系统的功能。组合逻辑控制方式主要用于高速计算机或小规模计算机中。

微程序控制方式是通过执行微指令来产生微命令的，主要优点如下：

① 设计规整，设计效率高；

② 易于修改、扩展指令系统功能；

③ 结构规整、简洁，可靠性高；

④ 性价比高。

其主要缺点是：产生微命令的速度慢，因为要多次访问控存读取微指令，访存速度限制了产生微命令的速度；另外，机器的执行效率不高，因为微指令格式较简单，没有充分发挥数据通路本身所具有的并行能力。微程序控制方式主要用于速度要求不高、功能较复杂的机器中，特别适合于系列机。

10．根据模型机数据通路结构，拟定 MOV 指令流程在 ST_2、ST_3、ST_4 中的操作时间表。

【答】

按主教材图 3-39 的 MOV 指令流程图，ST_2 完成一次内部数据通路操作，根据 ALU 输入选择、ALU 功能选择、输出移位选择、结果分配等四段操作设置相应的微命令。ST_2 结束后，若寻址方式为自增型寄存器间址，则进入目的周期 DT 或执行周期 ET；若寻址方式为自增型双间址或变址，则继续 ST，进入 ST_3，完成一次内部数据通路操作，然后进入 ST_4，完成一次从存储器读出，并经内部数据通路传送的操作。ST_2、ST_3、ST_4 中的操作时间表如表 3-3-3 所示。

表 3-3-3　MOV 指令源周期操作时间表

	电位型微命令	脉冲型微命令
ST_2：	R_i→A	CP R_i
	SP→A	CP SP
	PC→A	CP PC
	A+1	CP T　($\overline{P}$)
	DM	CP FT　($\overline{P}$)
	1→ST [@(R)+ ∨ X(R)]	CP ST　($\overline{P}$)
	T+1 [@(R)+ ∨ X(R)]	CP DT　($\overline{P}$)
	1→DT [(R)+] $\overline{DR}$	CP ET　($\overline{P}$)
	1→ET [(R)+] DR	
ST_3：	C→A	CP MAR
	R_i→B	CP T　($\overline{P}$)
	PC→B	
	输出 A [@(R)+]	
	A+B [X(R)]	
	DM	
	T+1	
ST_4：	EMAR	CP C
	R	CP T　($\overline{P}$)
	SMDR	CP FT　($\overline{P}$)
	MDR→B	CP ST　($\overline{P}$)
	输出 B	CP DT　($\overline{P}$)
	DM	CP ET　($\overline{P}$)
	1→DT 或	
	1→ET	

注：上述操作时间表中基本上未考虑逻辑条件。

11．拟出下述指令流程及操作时间表。

（1）MOV　(R_0), (SP)+;

（2）MOV　(R_1) +, X(R_0);

（3）MOV　R_2，(PC)+;

（4）MOV　−(SP), (R_3);

（5）ADD　R_1, X(R_0);

（6）SUB　(R_1)+, (R_2);

（7）AND　−(R_0), R_1;

（8）OR　R_2, (R_0)+;

（9）EOR　(R_0), (R_1);

（10）INC　X(PC);

（11）DEC　(R_0);

（12）COM　(R_1)+

（13）NEG　−(R_2);

（14）SL　R_0;

（15）SR　R_3;

（16）JMP　SKP;

（17）JMP　R_0;

（18）JMP　X(PC);

（19）RST(SP)+;

（20）JSR(R_1);

【答】

拟定指令流程关键是要了解模型机各类指令功能，掌握模型机各种寻址方式，熟悉模型机数据通路结构。此外，还需清楚指令中源地址和目的地址的位置，教材中约定，源在后，目的在前。

（1）MOV　(R_0), (SP)+;

该指令完成出栈操作。需要注意的是，在操作时间表中，当进行周期转换时，5 个打入命令都发，以便简化逻辑条件。另外，为简单起见，没有分开列出电位型微命令和脉冲型微命令。

1）指令流程	2）操作时间表
FT：M→IR PC+1→PC	FT：EMAR、R、SIR、 PC→A、A+1、DM、CPPC、1→ST、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)；
ST：SP→MAR M→MDR→C SP+1→SP	ST：SP→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)， EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$)， SP→A、A+1、DM、CPSP、1→DT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)；

DT：R_0→MAR	DT：R_0→A、输出 A、DM、CPMAR、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)；
ET：C→MDR MDR→M PC→MAR	ET：C→A、输出 A、DM、CPMDR、T+1、CPT($\overline{P}$)， EMAR、W、T+1、CPT($\overline{P}$)， PC→A、输出 A、DM、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（2）MOV　(R_1)+, X(R_0);

该指令将采用变址方式获得的源操作数送入目的单元。由于模型机指令字长有限，变址所需的形式地址放在现行指令所在单元的下一个单元中，取指后由 PC 指示。

1）指令流程	2）操作时间表
FT：M→IR PC+1→PC	FT：微命令同前；
ST：PC→MAR M→MDR→C	ST：PC→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$) EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$)
C+ R_0→MAR M→MDR→C PC+1→PC	C→A、R_0→B、A+B、DM、CPMAR、T+1、CPT($\overline{P}$)， EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$) PC→A、A+1、DM、CPPC、1→DT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
DT：R_1→MAR R_1+1→R_1	DT：R_1→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$) R_1→A、A+1、DM、CP R_1、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)；
ET：C→MDR MDR→M PC→MAR	ET：微命令同前 微命令同前 微命令同前

（3）MOV　R_2, (PC)+;

该条指令的源采用立即寻址，指令的功能是将立即数送入 R_2。与变址中的形式地址一样，16 位立即数存放在现行指令所在单元的下一个单元中，取指后由 PC 指示。

1）指令流程	2）操作时间表
FT：M→IR PC+1→PC	FT：微命令同前；
ST：PC→MAR M→MDR→C PC+1→PC	ST：PC→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$) EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$) PC→A、A+1、DM、CPPC、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
ET：C→R_2 PC→MAR	ET：C→A、输出 A、DM、CPR_2、T+1、CPT（P） PC→A、输出 A、DM、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（4）MOV　-(SP), (R_3);

该指令执行入栈操作。

1）指令流程	2）操作时间表
FT：M→IR	FT：微命令同前；
PC+1→PC	
ST：R_3→MAR	ST：R_3→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)
M→MDR→C	EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、1→DT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
DT：SP-1→SP、MAR	DT：SP→A、A-1、DM、CPSP、CPMAR、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
ET：C→MDR	ET：微命令同前
MDR→M	微命令同前
PC→MAR	微命令同前

（5）ADD　R_1, X(R_0);

该条指令的功能是将变址获得的源操作数与目的操作数相加，结果存放在目的地址 R_1 中。

1）指令流程	2）操作时间表
FT：M→IR	FT：微命令同前
PC+1→PC	
ST：PC→MAR	ST：微命令同前
M→MDR→C	微命令同前
C+ R_0→MAR	微命令同前
M→MDR→C	微命令同前
PC+1→PC	PC→A、A+1、DM、CPPC、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
ET：C+ R_1→R_1	ET：C→A、R_1→B、A+B、DM、CPR_1、T+1、CPT($\overline{P}$)
PC→MAR	PC→A、输出 A、DM、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（6）SUB　(R_1)+, (R_2);

该指令的功能是用源操作数减去目的操作数，结果存放在目的单元中。

1）指令流程	2）操作时间表
FT：M→IR	FT：微命令同前
PC+1→PC	
ST：R_2→MAR	ST：R_2→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)
M→MDR→C	EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、1→DT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)
DT：R_1→MAR	DT：R_1→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)
M→MDR→D	EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、T+1、CPT($\overline{P}$)

R_1+1→R_1　　R_1→A、A+1、DM、CPR_1、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：C-D→MDR　　ET：C→A、D→B、A-B、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　EMAR、W、T+1、CPT($\overline{P}$)

PC→MAR　　PC→A、输出 A、DM、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（7）AND　-(R_0), R_1;

该指令将两数相与，由于源采用寄存器寻址，取指后进入目的周期。

1）指令流程　　2）操作时间表

FT：M→IR　　FT：只需将 1→ST 改为 1→DT，其余微命令同前

PC+1→PC

DT：R_0-1→R_0、MAR　　DT：R_0→A、A-1、DM、CPR_0、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：R_1∧D→MDR　　ET：R_1→A、D→B、A∧B、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　EMAR、W、T+1、CPT($\overline{P}$)

PC→MAR　　微命令同前

（8）OR　R_2, (R_0)+;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ST：R_0→MAR　　ST：R_0→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→C　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$)

R_0+1→R_0　　R_0→A、A+1、DM、CPR_0、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：C∨R_2→R_2　　C→A、R_2→B、A∨B、DM、CPR_2、T+1、CPT($\overline{P}$)

PC→MAR　　微命令同前

（9）EOR　(R_0), (R_1);

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ST：R_1→MAR　　ST：R_1→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→C　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、1→DT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

DT：R_0→MAR　　DT：R_0→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：C EOR D→MDR　　ET：C→A、D→B、A EOR B、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　微命令同前

PC→MAR　　微命令同前

（10）INC　X(PC);

该加 1 指令采用相对寻址，位移量放在现行指令所在单元的下一单元中，取指后由 PC 指示。

1）指令流程　　2）操作时间表

FT：M→IR　　FT：EMAR、R、SIR、PC→A、A+1、DM、CPPC、1→DT、

PC+1→PC　　CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

DT：PC→MAR　　DT：PC→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、T+1、CPT($\overline{P}$)

PC+D→MAR　　PC→A、D→B、A+B、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、T+1、CPT($\overline{P}$)

PC+1→PC　　PC→A、A+1、DM、CPPC、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：D+1→MDR　　ET：D→A、A+1、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　微命令同前

PC→MAR　　微命令同前

在模型机的相对寻址中，基准地址 PC 是指向位移量，而不是指向下条指令，因此在 DT 中，PC+1→PC 的操作应在变址计算 PC+D→MAR 的操作之后进行。

（11）DEC　(R_0);

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

DT：R_0→MAR　　DT：R_0→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：D-1→MDR　　ET：D→A、A-1、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　微命令同前

PC→MAR　　微命令同前

（12）COM（R_1）+

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

DT：R_1→MAR　　DT：R_1→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、T+1、CPT($\overline{P}$)

R_1+1→R_1　　R_1→A、A+1、DM、CPR$_1$、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：D→MDR　　ET：D→A、输出 A、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　微命令同前

PC→MAR　　微命令同前

（13）NEG　-(R_2);

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

DT：R_2-1→R_2、MAR　DT：R_2→A、A-1、DM、CPR_2、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→D　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPD、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：D+1→MDR　　ET：D→A、A+1、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　微命令同前

PC→MAR　　微命令同前

（14）SL　R_0;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：将 1→DT 改为 1→ET，其余微命令同前；

PC+1→PC

ET：$\overset{\frown}{R}_0$→R_0　　ET：$\overset{\frown}{R}_0$→A、输出 A、DM、CPR_0、T+1、CPT($\overline{P}$)，

PC→MAR　　微命令同前。

（15）SR　R_3;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前；

PC+1→PC

ET：$\overset{\frown}{R}_3$→R_3　　ET：$\overset{\frown}{R}_3$→A、输出 A、DM、CPR_3、T+1、CPT($\overline{P}$)，

PC→MAR　　微命令同前。

（16）JMP　SKP;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ET：PC+1→PC、MAR　ET：PC→A、A+1、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（17）JMP　R_0;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ET：R_0→PC、MAR　　ET：R_0→A、输出 A、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（18）JMP　X（PC）;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ET：PC→MAR　　ET：PC→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→C　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、T+1、CPT($\overline{P}$)

PC+C→PC、MAR　　PC→A、C→B、A+B、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

转移指令采用相对寻址时，在进行了变址计算后，不再做 PC+1→PC 的操作。这是因为转移指令执行后要转到转移地址去执行目标指令，而不是顺序执行紧跟转移指令的下条指令。

（19）RST　(SP)+;

1）指令流程　　2）操作时间表

FT：M→IR　　FT：微命令同前

PC+1→PC

ET：SP→MAR　　ET：SP→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

SP+1→SP　　SP→A、A+1、DM、CPSP、T+1、CPT($\overline{P}$)

M→MDR→PC、MAR　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

（20）JSR　(R_1);

转子指令在取指后进入源周期获取子程序入口，再进入执行周期保存断点，并转到子程序入口。

1）指令流程　　2）操作时间表

FT：M→IR　　FT：将 1→ET 改为 1→ST，其余微命令同前

PC+1→PC

ST：R_1→MAR　　ST：R_1→A、输出 A、DM、CPMAR、T+1、CPT($\overline{P}$)

M→MDR→C　　EMAR、R、SMDR、MDR→B、输出 B、DM、CPC、1→ET、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

ET：SP-1→SP、MAR　　ET：SP→A、A-1、DM、CPSP、CPMAR、T+1、CPT($\overline{P}$)

PC→MDR　　PC→A、输出 A、DM、CPMDR、T+1、CPT($\overline{P}$)

MDR→M　　EMAR、W、T+1、CPT($\overline{P}$)

C→PC、MAR　　C→A、输出 A、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

该题也可以采用另一种作法，即取指后进入执行周期，先保存断点，再获取子程序入口并转该入口。相应的指令流程如下：

FT：M→IR

PC+1→PC

ET：SP-1→SP、MAR
PC→MDR
MDR→M
R_1→MAR
M→MDR→PC、MAR

12．拟出中断周期IT中各拍的操作时间表。

【答】

假设采用模型机中断机制。当 CPU 响应中断后，向中断控制器发出中断响应信号 INTA，并进入中断周期。中断控制器收到 INTA 后，将被批准的中断源的中断号送往数据总线，以便 CPU 接收。在中断周期中，CPU 关中断，保存断点，将中断号乘以 2，作为向量地址访问向量表，获得中断服务程序入口地址后，转入服务程序。

中断周期流程和操作时间表如下：

1）IT 流程	2）操作时间表
IT：关中断	IT：0→I
SP-1→SP、MAR	SP→A、A-1、DM、CPSP、CPMAR、T+1、CPT($\overline{P}$)
PC→MDR	PC→A、输出 A、DM、CPMDR、T+1、CPT($\overline{P}$)
MDR→M	EMAR、W、T+1、CPT($\overline{P}$)
8259→MDR→MAR	IOR、SMDR、MDR→B、输出 B、左移、CPMAR、T+1、CPT($\overline{P}$)
M→MDR→PC、MAR	EMAR、R、SMDR、MDR→B、输出 B、DM、CPPC、CPMAR、1→FT、CPT($\overline{P}$)、CPFT($\overline{P}$)、CPST($\overline{P}$)、CPDT($\overline{P}$)、CPET($\overline{P}$)

在关中断操作中，发 0→I 命令，将模型机程序状态字 PSW 中的允许中断位 I 清零。在 8259→MDR→MAR 操作中，发 IOR 命令，从 8259 读出中断号，经 MDR、ALU 传送，并将 ALU 输出的结果左移一位，即获得向量地址（中断号乘以 2）。

13．编写取目的地址微子程序（从 60H单元开始）。

【答】

取目的地址微子程序与取源操作数微子程序类似，所不同的是，前者只取出目的地址，而不取目的操作数。微子程序如表 3-3-4 所示。

14．根据表 3-13 微程序，以微地址序列形式（如 00-01-02-0C…），拟出下述指令的读出与执行过程。

（1）MOV　(R_0), (SP)+;
（2）MOV　(R_1)+, X(R_0);
（3）ADD　X(R_0), R_1;
（4）SUB　(R_1)+, (R_2);
（5）NEG　-(R_2);
（6）JMP　(R_0);
（7）JSR　R_1;

【答】

注意 MOV 指令和双操作数指令中源地址与目的地址的位置，源地址在后，目的地址在前。

表 3-3-4　取目的地址微子程序

	微地址	操作	微命令
取目的地址	60		按目的寻址方式分支，SC=0110
R	61		返回，SC=1000
(R)	62	R_j→MAR	R_j→A，输出 A，DM，CPMAR，返回，SC=1000
−(R)	63	R_j-1→R_j	R_j→A，A-1，DM，CPR_j，SC=0000
	64		转 62，SC=0001
(R)+	65	R_j→MAR	R_j→A，输出 A，DM，CPMAR，SC=0000
	66	R_j+1→R_j	R_j→A，A+1，DM，CPR_j，返回，SC=1000
@(R)+	67	R_j→MAR	R_j→A，输出 A，DM，CPMAR，SC=0000
	68	M→MDR→MAR	EMAR，R，SMDR，MDR→B，输出 B，DM，CPMAR，SC=0000
X(R)	69		转 66，SC=0001
	6A	PC→MAR	PC→A，输出 A，DM，CPMAR，SC=0000
	6B	PC+1→PC	PC→A，A+1，DM，CPPC，SC=0000
	6C	M→MDR→D	EMAR，R，SMDR，MDR→B，输出 B，DM，CPD，SC=0000
	6D	R_j+D→MAR	R_j→A，D→B，A+B，DM，CPMAR，返回，SC=1000

（1）MOV　(R_0), (SP)+;

00-01-02-03-4C-52-53-54-4F-04-60-62-05-06-07-08-09-00

解释微地址序列：先取指（00-01-02），再转 MOV 指令入口（03），然后转“取源操作数”入口（4C），进入自增型寄存器间址（52-53-54-4F），返回（04），再转“取目的地址”入口（60），进入寄存器间址（62），返回（05），最后执行传送操作并转取指入口（06-07-08-09-00）。

（2）MOV　(R_1)+, X(R_0);

00-01-02-03-4C-59-5A-5B-5C-5D-4F-04-60-65-66-05-06-07-08-09-00

解释微地址序列：取指（00-01-02），再转 MOV 指令入口（03），然后转“取源操作数”入口（4C），进入变址（59-5A-5B-5C-5D-4F），返回（04），再转“取目的地址”入口（60），进入自增型寄存器间址（65-66），返回（05），最后执行传送操作并转取指入口（06-07-08-09-00）。

（3）ADD　X(R_0), R_1;

00-01-02-0C-4C-4D-0D-60-6A-6B-6C-6D-0E-0F-10-11-07-08-09-00

解释微地址序列：ADD 是双操作数指令，因此取指后进入其入口（0C），转“取源操作数”入口（4C），进入寄存器寻址（4D），返回（0D），再转“取目的地址”入口（60），进入变址（6A-6B-6C-6D），返回（0E-0F），最后执行加法操作并转取指入口（10-11-07-08-09-00）。

（4）SUB　(R_1)+, (R_2);

00-01-02-0C-4C-4E-4F-0D-60-65-66-0E-0F-14-15-07-08-09-00

解释微地址序列：取指（00-01-02），进入双操作数指令入口（0C），转“取源操作数”入口（4C），进入寄存器间址（4E-4F），返回（0D），再转“取目的地址”入口（60），进入自增型寄存器间址（65-66），返回（0E-0F），最后执行减法操作并转取指入口（14-15-07-08-09-00）。

（5）NEG　-(R_2);

00-01-02-24-60-63-64-62-25-26-2B-2C-07-08-09-00

解释微地址序列：NEG 是单操作数指令，取指后进入其入口（24），转“取目的地址”入口（60），进入自减型寄存器间址（63-64-62），返回（25-26），最后执行求补操作并转取指入口（2B-2C-07-08-09-00）。

（6）JMP　(R_0);

00-01-02-3F-43-4C-4E-4F-44-45-08-09-00

解释微地址序列：JMP 是转移指令，取指后进入其入口（3F-43），转“取源操作数”入口（4C），进入寄存器间址（4E-4F），返回（44-45），将转移地址送入 MAR 并转取指入口（08-09-00）。

（7）JSR　R_1;

00-01-02-3F-46-48-49-4A-4B-47-43-4C-4D-44-45-08-09-00

解释微地址序列：JSR 是转子指令，取指后进入其入口（3F-46），转“压栈”入口，将转子时的返回地址压入堆栈保存（48-49-4A-4B），返回（44-45），将子程序入口地址送入 MAR 并转取指入口（08-09-00）。

15．如果以R_3为堆栈指针，软件建立堆栈，试分别编写压栈及弹出操作的子程序。

【答】

假设栈顶单元地址为 1000H，压栈和弹出之前，用立即寻址方式将 1000H 送入堆栈指针 R_3，即执行指令“MOV　R_3, (PC)+”（该指令所在单元的下一单元存放 1000H）。之后，将需要压栈的数据所在单元的地址送入某一寄存器，如 R_0，再调用压栈子程序，该子程序包含一条传送指令：

MOV　-(R_3), (R_0);　　将 R_0 所指示的存储单元中的数据压入堆栈。

若需要弹出操作，则调用弹出子程序，该子程序也包含一条传送指令：

MOV　(R_1), (R_3)+;　将数据从堆栈弹出，送入 R_1 所指示的存储单元。

16．如果将CPU时钟周期与访存周期分开设置，一个访存周期占用 4 个时钟周期，试重新设计模型机指令流程。

【答】

一个访存周期占用 4 个时钟周期，可在第一个时钟周期中传送地址，即 CPU 将地址从 MAR 送往存储器；在第二个时钟周期中向存储器发读令或写令；第三个时钟周期从存储器读出数据或向存储器写入数据；第四个时钟周期结束存储器访问，即 CPU 撤销地址、命令、数据。

下面，以加法指令“ADD　(R_1), R_0”为例，重新拟定模型机指令流程。

FT0：MAR→M　　;进入 FT，将现行指令地址经地址总线送往存储器

FT1：读令→M　　;将读命令经控制总线送往存储器

FT2：M→IR　　;从存储器读出指令，经数据总线送往 IR

FT3：PC+1→PC　　;撤销总线上的地址、命令、指令，并修改 PC，结束 FT

DT0：R_1→MAR　　;进入 DT，将 R_1 内容送往 MAR

DT1：MAR→M　　;将目的地址经地址总线送往存储器

DT2：读令→M　　;将读命令经控制总线送往存储器
DT3：M→MDR　　;从存储器读出目的操作数，经数据总线送往 MDR
DT4：MDR→D　　;撤销总线上的地址、命令、数据，并将目的操作数从 MDR 送
;往暂存器 D，结束 DT
ET0：R_0+D→MDR　　;进入 ET，将源操作数和目的操作数相加，结果送往 MDR
ET1：MAR→M　　;将目的地址经地址总线送往存储器
ET2：写令→M　　;将写命令经控制总线送往存储器
ET3：MDR→M　　;将运算结果经数据总线写入存储器
ET4：PC→MAR　　;撤销总线上的地址、命令、数据，并将下条指令的地址送
;往 MAR

17．如果将模型机内部数据通路结构改为图 3.3.11 结构，试重新设计模型机。

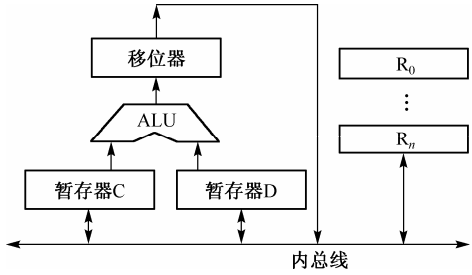


图 3.3.11　采用集成寄存器结构的 CPU 数据通路

【答】

通用寄存器组采用存储器结构，输入逻辑采用锁存器，内总线采用双向形式，其他寄存器均挂接在内总线上。模型机数据通路结构框图如图 3.3.12 所示。

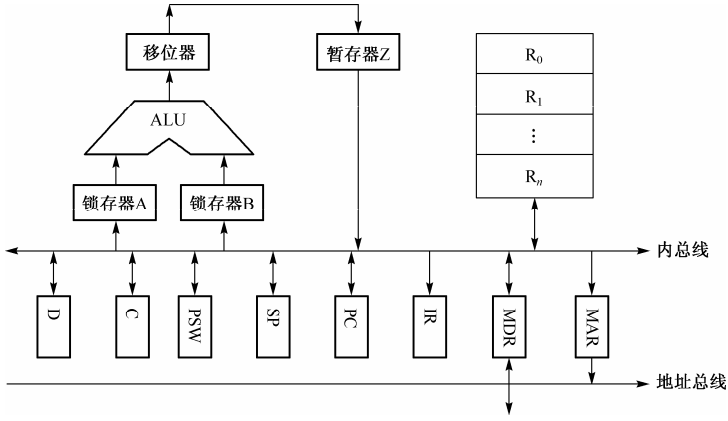


图 3.3.12　新设计的模型机

3.4 第 4 章习题解答及解题思路

3.4.1 第 4 章重点和难点解析

在本章学生需掌握的内容中，半导体存储器的逻辑设计方法涉及的内容较广，变化也最多。在半导体存储器的逻辑设计中，涉及四个层次的内容：其一是芯片的选择，这需要根据所需设计的存储器的容量（包括单元数×位数）和已给的存储器芯片的规格，来确定每种规格存储器芯片的数量；其二是要确定寻址系统，这需要根据设计的存储器容量和各芯片的容量，分别确定存储系统的地址范围和各芯片的地址范围；其三是在以上基础上设计芯片片选信号的逻辑；最后要确定存储芯片与地址线，数据线，控制信号线和片选信号线之间的连接逻辑。在每个层次的内容中，又可以产生不同的设计变化形式，如根据存储容量确定地址范围，可以是已知容量确定地址范围和连接的地址信号线数，也可能是已知将要设计的存储系统安排在一特定的地址范围内，再来确定其容量和连接的地址线数等。又如，片选信号的逻辑设计，可以是只考虑主存储器各芯片的片选逻辑，也可能需要考虑主存和 I/O 统一编址的情况下的片选逻辑，还可能需要考虑当前读写主存和 I/O 的控制信号的参与下的片选信号的设计逻辑等。以下就主要针对存储器的逻辑设计方法所涉及的各方面的问题举例说明其分析和解决的方法。

1．若某机字长 32 位，主存储器容量为 256MB，问：

（1）若按字节编址，则地址寄存器，数据寄存器各为多少位？编址范围是多大？

按字节编址，每个编址单元存放一个字节代码。

256MB=2^n×8 位，n = 28，地址寄存器为 28 位、数据寄存器为 8 位，编址范围是 0000000H～FFFFFFF H。

（2）若按半字编址，则地址寄存器、数据寄存器各为多少位？编址范围是多大？

按半字编址，每个编址单元存放 16 位二进制代码：

256 MB = 2^{28} ×8 位 = 2^{27} × 16 位，地址寄存器为 27 位，数据寄存器为 16 位，编址范围是：0000000H～7FFFFFFH。

（3）若按字编址，则地址寄存器、数据寄存器各为多少位？编址范围是多大？

按字编址，每个编址单元存放 32 位二进制代码。

256 MB = 2^{28}×8 位 = 2^{26} ×32 位，地址寄存器为 26 位，数据寄存器为 32 位，编址范围是：0000000H × 3FFFFFFH。

2．某 8 位机的主存容量为 64 KB，用 4 片RAM芯片组成（16 KB/片），地址线A_{15}～A_0（低位），请列出各片选信号的逻辑式。

64 KB 容量其地址为：$\underline{A_{15}A_{14}}$　　$\underline{A_{13}A_{12}\text{L}\ A_0}$（低）

片选地址　16 KB 的芯片地址

$A_{15}A_{14}$ 经 2-4 译码后作为片选信号：

$0^{\#}$ RAM 芯片：$CS_0 = \overline{A}_{15}\overline{A}_{14}$

$1^{\#}$ RAM 芯片：$CS_1 = \overline{A}_{15}A_{14}$

$2^{\#}$ RAM 芯片：$CS_2 = A_{15}\overline{A}_{14}$

$3^{\#}$ RAM 芯片：$CS_3 = A_{15}A_{14}$

3. 若某 8 位机主存容量 7 KB，用 3 块RAM存储芯片（分别为 4 KB/片、2 KB/片、1 KB/片）组成，地址线为A_{15}～A_0（低），试写出 3 个芯片的片选信号的逻辑式。

主存容量 7 KB，故地址寄存器应为 13 位（按 8 K 单元计算），即主存地址线为 A_{12}～A_0（低），地址线 $A_{15}A_{14}A_{13}$ 不用，主存编址范围是 0000H～1BFFH。用题中所给芯片组成 7 KB 的主存时，有 6 种空间分配方案，如图3.4.1 所示。

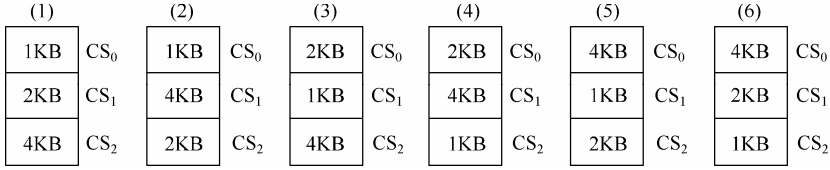


图 3.4.1　存储空间分配图

我们可以参照图 3.4.1，写出各空间分配方案时各芯片的片选信号逻辑式。

方案（1）1 KB 的芯片，地址 $A_{12}A_{11}A_{10}$ 为 000，故 $CS_0 = \overline{A}_{12}\overline{A}_{11}\overline{A}_{10}$

2 KB 的芯片，地址 $A_{12}A_{11}A_{10}$ 为 001～010，故 $CS_1 = \overline{A}_{12}\overline{A}_{11}A_{10} + \overline{A}_{12}A_{11}\overline{A}_{10} = A_{12}(A_{11} \oplus A_{10})$

4 KB 的芯片，地址 $A_{12}A_{11}A_{10}$ 为 011～110，故 $CS_2 = \overline{A}_{12}A_{11}A_{10} + A_{12}\overline{A}_{11}\overline{A}_{10} + \overline{A}_{12}A_{11}A_{10} + A_{12}A_{11}\overline{A}_{10}$

方案（2）$CS_0 = \overline{A}_{12}\overline{A}_{11}\overline{A}_{10}$

$CS_1 = \overline{A}_{12}\overline{A}_{11}A_{10} + \overline{A}_{12}A_{11}\overline{A}_{10} + \overline{A}_{12}A_{11}A_{10} + A_{12}\overline{A}_{11}\overline{A}_{10}$

$CS_2 = A_{12}\overline{A}_{11}A_{10} + A_{12}A_{11}\overline{A}_{10}$

方案（3）$CS_0 = \overline{A}_{12}\overline{A}_{11}\overline{A}_{10} + \overline{A}_{12}\overline{A}_{11}A_{10} + \overline{A}_{12}\overline{A}_{11}$

$CS_1 = \overline{A}_{12}A_{11}\overline{A}_{10}$

$CS_2 = \overline{A}_{12}A_{11}A_{10} + A_{12}\overline{A}_{11}\overline{A}_{10} + A_{12}\overline{A}_{11}A_{10} + A_{12}A_{11}\overline{A}_{10}$

方案（4）$CS_0 = \overline{A}_{12}\overline{A}_{11}$

$CS_1 = \overline{A}_{12}A_{11}\overline{A}_{10} + \overline{A}_{12}A_{11}A_{10} + A_{12}\overline{A}_{11}\overline{A}_{10} + A_{12}\overline{A}_{11}A_{10}$

$CS_2 = A_{12}A_{11}\overline{A}_{10}$

方案（5）$CS_0 = \overline{A}_{12}\overline{A}_{11}\overline{A}_{10} + \overline{A}_{12}\overline{A}_{11}A_{10} + \overline{A}_{12}A_{11}\overline{A}_{10} + \overline{A}_{12}A_{11}A_{10} = \overline{A}_{12}$

$CS_1 = A_{12}\overline{A}_{11}\overline{A}_{10}$

$CS_2 = A_{12}\overline{A}_{11}A_{10} + A_{12}A_{11}\overline{A}_{10} = A_{12}(A_{11} \oplus A_{10})$

方案（6）$CS_0 = A_{12}$

$CS_0 = A_{12}\overline{A}_{11}\overline{A}_{10} + A_{12}\overline{A}_{11}A_{10} = A_{12}\overline{A}_{11}$

$CS_2 = A_{12}A_{11}\overline{A}_{10}$

在以上方案中，显然方案 6 最简单。

4．某存储器容量 64 KB，按字节编址，若要使每页有 256 个单元，试问可分为多少页？并以十六进制给出页面地址范围。

64 KB，每页 256 个单元，所以页数为 64K ÷ 256 = 256 页，64 KB，地址码应为：

$A_{15}A_{14}A_{13}A_{12}$	$A_{11}A_{10}A_9A_8$	A_7L A_0
0H	0H	页内地址
M	M	
FH	FH	

所以，页面地址范围应为 00～FFH。

5．若 CPU 送出的地址线为 A_{15}～A_0（低位），数据线 D_7～D_0，读写控制信号为 R / $\overline{W}$ 。

（1）用 1 K×8 位/片的 RAM 芯片设计一个 4 KB 的主存，画出存储器逻辑图，注明各信号线，列出片选逻辑式。

RAM 芯片容量为 1K×8 位（即 1KB），用 4 片作字扩展构成 4KB 的主存。主存地址应为 12 位（A_{11}～A_0），1K 的芯片地址（即片内地址）为 A_9～A_0，故取主存高位地址 $A_{11}A_{10}$ 作为片选地址，译码产生 $CS_0 = \overline{A}_{11}\overline{A}_{10}$，$CS_1 = A_{11}\overline{A}_{10}$，$CS_2 = A_{11}\overline{A}_{10}$，$CS_3 = A_{11}A_{10}$。

存储器以及与 CPU 的连接如图3.4.2 所示。

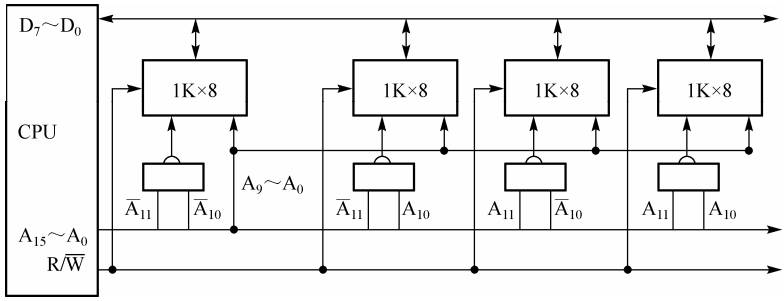


图 3.4.2　存储器与 CPU 连接图（1）

（2）用 4 K×4 位/片的 RAM 芯片设计一个 4 KB 的主存，画出芯片级逻辑图，注明各信号线，列出片选逻辑式。

RAM 芯片容量为 4 KB×4 位/片，用两片拼接进行位扩展，可构成 4 K×8 位（即 4 KB）的主存，其逻辑图如图3.4.3 所示，图中 MREQ 作为芯片的片选信号。

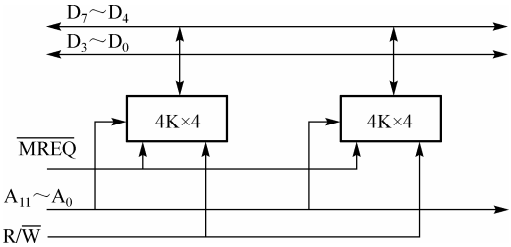


图 3.4.3　存储器与 CPU 连接图（2）

6．若某机的地址总线为 A_{15}～A_0（低），数据总线为 D_7～D_0，读/写控制信号为 R/$\overline{W}$ 。

（1）选用 1 K×8 位/片的 RAM 芯片设计一个 4 K×16 位的存储器，画出存储器芯片级逻辑图，注明各信号线，写出片选信号逻辑式。

用 2 片 1K×8 位/片的 RAM 芯片拼接进行位扩展，构成一个 1K×16 位的存储模块，使用同一个片选信号，共需 4 个存储模块即可构成 4 K×16 位的存储器。

地址分析：

$A_{15}A_{14}A_{13}A_{12}$	$A_{11}A_{10}$	A_9～A_0
不用	片选信号	芯片地址
	4 K×16 位存储器地址	

片选信号逻辑：$CS_0=\overline{A}_{11}\overline{A}_{10}$，$CS_1=\overline{A}_{11}A_{10}$，$CS_2=A_{11}\overline{A}_{10}$，$CS_3=A_{11}A_{10}$。

存储器逻辑图如图3.4.4所示。

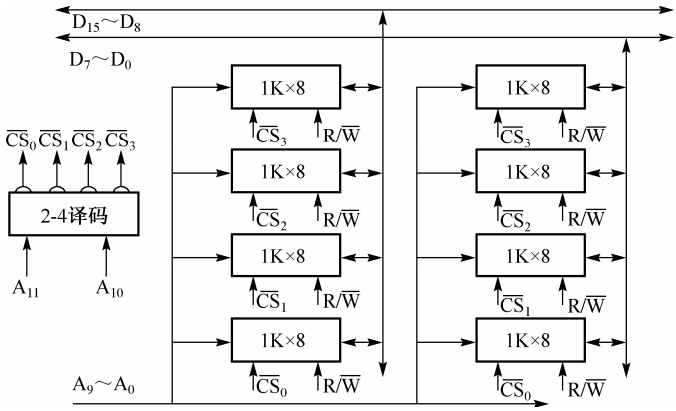


图 3.4.4　存储器连接逻辑图

（2）选用 2 K×8 位/片的 RAM 芯片设计一个 8 K×16 位的存储器，画出存储器芯片级逻辑图，注明各信号线，写出片选信号逻辑式。

用两片 2 K×8 位的 RAM 芯片拼接进行位扩展，构成一个 2 K×16 位的存储模块，共需 4 个存储模块即可构成 8 K×16 位的存储器。

地址分析：

$A_{15}A_{14}A_{13}$	$A_{12}A_{11}$	A_{10}～A_0
不用	片选地址	2 K 芯片地址
	8 K 的存储器地址	

片选逻辑：$CS_0=\overline{A}_{12}\overline{A}_{11}$，$CS_1=\overline{A}_{12}A_{11}$，$CS_2=A_{12}\overline{A}_{11}$，$CS_3=A_{12}A_{11}$。

存储器逻辑图与（1）题图的结构一致，只是将图中的 2-4 译码器输入端的 $A_{11}A_{10}$ 改为 $A_{12}A_{11}$，地址线 A_9～A_0 改为 A_{10}～A_0，芯片 1K×8 改为 2K×8 即可。

（3）选用 4K×8 位/片的 RAM 芯片设计一个 16K×16 位的存储器，画出存储器芯片级逻辑图，注明各信号线，写出片选信号逻辑式。

做法与（1）、（2）类似。

地址分析：

$A_{15}A_{14}$	$A_{13}A_{12}$	A_{11} A_{10} ～ A_0
不用	片选地址	4 K 芯片地址
	16 K 存储器地址	

片选信号逻辑式：$CS_0=\overline{A}_{13}\overline{A}_{12}$，$CS_1=\overline{A}_{13}A_{12}$，$CS_2=A_{13}\overline{A}_{12}$，$CS_3=A_{13}A_{12}$。

存储器逻辑图与图（1）的结构一致，只是将图中的 2–4 译码器输入 $A_{11}A_{10}$ 改为 $A_{13}A_{12}$，地址线 A_9～A_0 改为 A_{11}～A_0，芯片 1K×8 改为 4K×8 即可。

（4）选用 8K×8 位/片的 RAM 芯片设计一个 32K×16 位的存储器，画出存储器芯片级逻辑图，注明各信号线，写出片选信号逻辑式。

做法与（1）类似。

地址分析：

A_{15}	A_{14} A_{13}	A_{12} A_{11} A_{10} ～ A_0
不用	片选地址	8 K 芯片地址
	32 K 存储器地址	

片选逻辑：$CS_0=\overline{A}_{14}\overline{A}_{13}$，$CS_1=\overline{A}_{14}A_{13}$，$CS_2=A_{14}\overline{A}_{13}$，$CS_3=A_{14}A_{13}$。

存储器逻辑图与图 3.4.4 一致，只是将图中的 2–4 译码器输入 $A_{11}A_{10}$ 改为 $A_{14}A_{13}$，地址线 A_9～A_0 改为 A_{12}～A_0，芯片 1K×8 改为 8K×8 即可。

（5）选用 16K×8 位/片的 RAM 芯片设计一个 64K×16 位的存储器，画出存储器芯片级逻辑图，注明各信号线，写出片选信号逻辑。

做法与（1）类似。

地址分析：

$A_{15}A_{14}$	$A_{13}A_{12}$ A_{11} A_{10}～A_0
片选地址	16 K 芯片地址
64 K 存储器地址	

片选逻辑：$CS_0=\overline{A}_{15}\overline{A}_{14}$，$CS_1=\overline{A}_{15}A_{14}$，$CS_2=A_{15}\overline{A}_{14}$，$CS_3=A_{15}A_{14}$。

存储器逻辑图与图3.4.4一致，只是将图中的 2–4 译码器输入 $A_{11}A_{10}$ 改为 $A_{15}A_{14}$，地址线 A_9～A_0 改为 A_{13}～A_0，芯片 1 K×8 改为 16 K×8 即可。

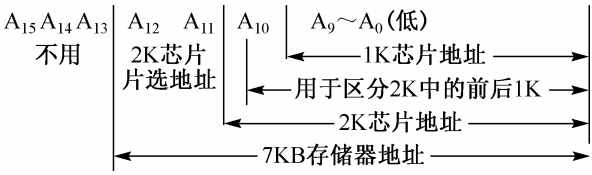
7. 试设计一个容量为 7 KB 的半导体存储器，其中固化区 4 KB，选用 EPROM 芯片 2716（2 K×8 位/片）；随机读写区 3 KB，可选用 SRAM 芯片：2 K×4 位/片和 1 K×4 位/片。地址总线为 A_{15}～A_0（低），双向数据总线 D_7～D_0（低），读/写控制信号为 R / $\overline{W}$，访存请求信号为 $\overline{MREQ}$（为低，则允许访存），请画出存储器芯片级逻辑图，注明各信号线，写出各芯片的片选信号逻辑式。

（1）设置存储空间分配图，确定芯片数量。

假设存储空间低地址区为固化区，高地址区为随机读写区，则空间分配如下：

$EPROM_0$	EPROM	2 KB	固化区
$EPROM_1$	EPROM	2 KB	（4K×8）
$SRAM_0$ 模块	SRAM 2K×4 位	SRAM 2K×4 位	随机读写区
$SRAM_1$ 模块	SRAM 1K×4 位	SRAM 1K×4 位	（3K×8）

（2）地址分配与片选信号如下：



所设计的存储器是 7 KB，共 7 K 个编址单元，故所需地址线为 13 条（A_{12}～A_0），地址线的 A_{15}～A_{13} 始终为 0，可略去不用；容量为 2 K 的芯片，其地址线应有 11 条（A_{10}～A_0），容量为 1K 的芯片，其地址线 10 条（A_9～A_0），故 2 K 的芯片片选信号由 $A_{12}A_{11}$ 的状态来定，用 A_{10} 来区分 2 K 中的前后 1 K。这样，芯片、地址、片选的关系如下：

芯片（或模块）	容量	芯片地址	片选逻辑
$EPROM_0$	2 KB	A_{10}～A_0	$CS_0 = \overline{A}_{12}\overline{A}_{11}$
$EPROM_1$	2 KB	A_{10}～A_0	$CS_1 = \overline{A}_{12}A_{11}$
$SRAM_0$	2 KB	A_{10}～A_0	$CS_2 = A_{12}\overline{A}_{11}$
$SRAM_1$	1 KB	A_9～A_0	$CS_3 = A_{12}A_{11}\overline{A}_{10}$

（3）画出逻辑图

逻辑图如图3.4.5所示。

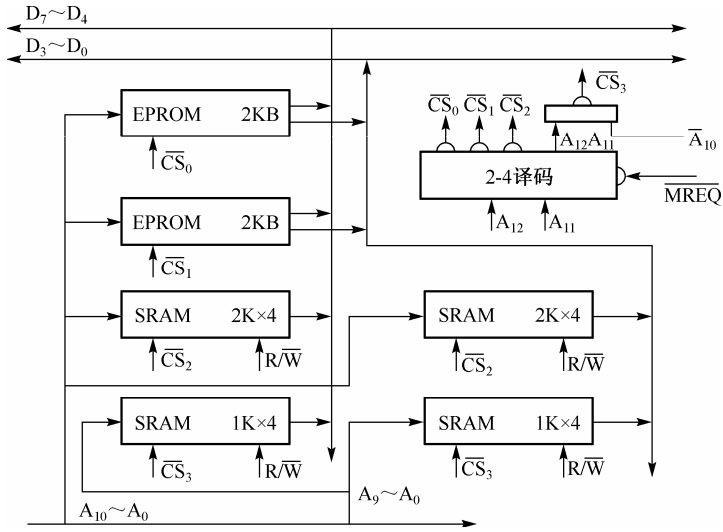


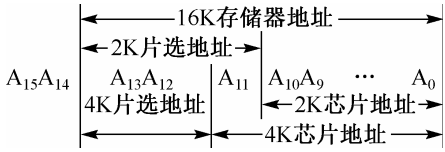
图 3.4.5　例题 7 存储器芯片级逻辑图

8．设计一个容量为 16 KB（按字节编址）的存储器，从低地址向高地址依次为 4 KB 固化区，选用 2 KB/片的 EPROM 芯片；4 KB 的空区（无存储芯片）；8 KB 的随机读写区，选用 4 K×4 位的 RAM 芯片。CPU 的地址总线为 A_{15}～A_0（低），双向数据总线 D_7～D_0（低），读/写控制信号为 R / $\overline{W}$，访存请求信号为 $\overline{MREQ}$，请设计出该存储器，并与 CPU 连接。

固化区 4 KB 需要 2 片 2 KB 的 EPROM 芯片，用 2 片 4 K×4 位的 RAM 芯片拼接作位扩展，构成一个 4 KB 的存储模块。用 3 个这样的存储模块作字扩展即构成 8 KB 的随机读写区，

存储器空间分布如图3.4.6所示。

地址分配分析如下：



A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	…	A_1	A_0		
0	0	0	0	0	0	…	0	0	2 KB（EPROM）	CS_0
				M						
0	0	0	0	0	**1**	…	1	1		
0	0	0	0	**1**	0	…	0	0	2 KB（EPROM）	CS_1
				M						
0	0	0	0	1	1	…	1	1		
0	0	0	**1**	0	0	…	0	0		CS_2
				M						
0	0	0	1	1	1	…	1	1		
0	0	**1**	0	0	0	…	0	0	4 K×4 位（RAM） 4 K×4 位（RAM）	CS_3
				M						
0	0	1	0	1	1	…	1	1		
0	0	**1**	1	0	0	…	0	0	4 K×4 位（RAM） 4 K×4 位（RAM）	CS_4
				M						
0	0	1	1	1	1	…	1	1		

图 3.4.6　存储空间分布图

（1）选用 2-4 译码器产生片选信号的存储器实现方案

这种方案是将存储器空间分为 4 个 4 KB，这样需要 4 个片选信号，其中固化区由 2 片 2 KB 芯片构成，故需要用 A_{11} 来区分 4 KB 中的前后 2 KB，这是以芯片地址线多的为主进行译码的方案，于是片选信号逻辑式如下：

$$CS_0 = \overline{A}_{13}\overline{A}_{12}\overline{A}_{11}$$
$$CS_1 = \overline{A}_{13}\overline{A}_{12}A_{11}$$
$$CS_2 = \overline{A}_{13}A_{12}$$
$$CS_3 = A_{13}\overline{A}_{12}$$
$$CS_4 = A_{13}A_{12}$$

存储器固化区是只读不写的，它一工作就读取数据，故加片选信号使它选中，而不加读写控制信号。

其芯片连接图如图3.4.7所示。

（2）选用 3-8 译码器产生片选信号的存储实现方案

这是以芯片地址线少的为主，进行译码的方案。把存储器空间看成 8 个 2 KB 空间构成，于是用 $A_{13}A_{12}A_{11}$ 作为 2 KB 芯片的片选地址进行译码，对于 4 KB 的存储模块，则需要两个相应输出信号相或后来进行片选，于是片选信号逻辑式如下：

$$CS_0 = \overline{A}_{13}\overline{A}_{12}\overline{A}_{11}$$
$$CS_1 = \overline{A}_{13}\overline{A}_{12}A_{11}$$
$$CS_2 = \overline{A}_{13}A_{12}\overline{A}_{11} + \overline{A}_{13}A_{12}A_{11}$$
$$CS_3 = A_{13}\overline{A}_{12}\overline{A}_{11} + A_{13}\overline{A}_{12}A_{11}$$
$$CS_4 = A_{13}A_{12}\overline{A}_{11} + A_{13}A_{12}A_{11}$$

此方案实现的存储器结构图同方案（1）的图3.4.7，只要将图中片选信号产生电路用 3-8 译码电路代替即可。其芯片连接图如图3.4.8所示。

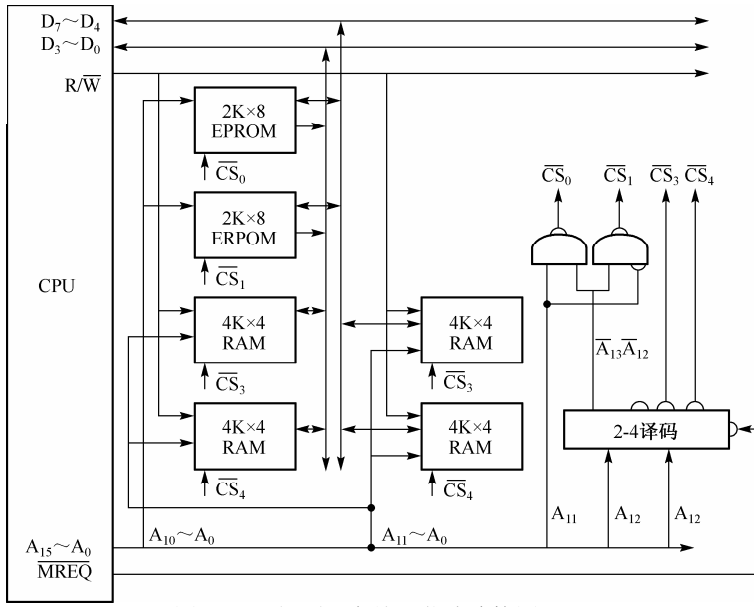


图 3.4.7　例题 8 存储器芯片连接图（1）

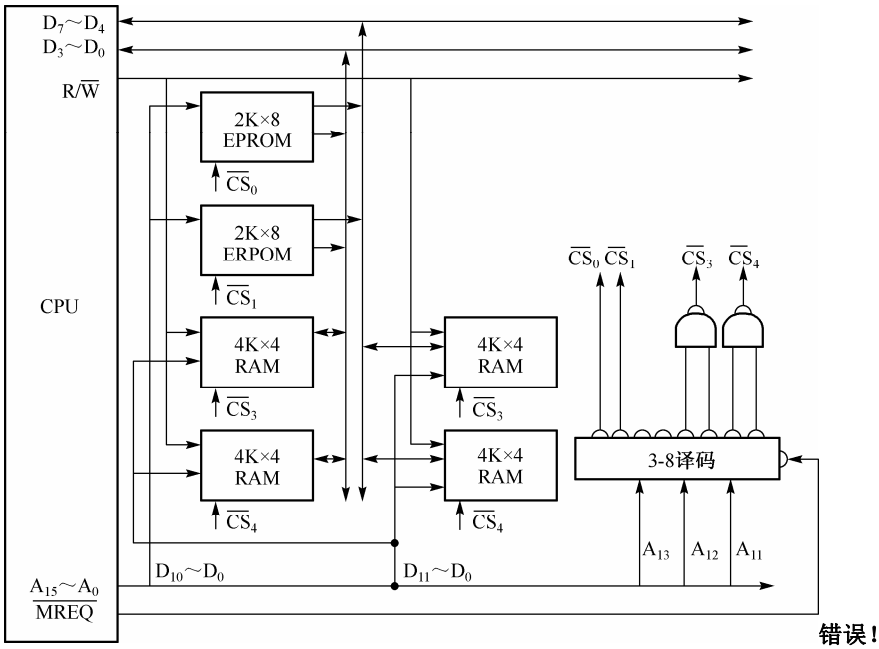


图 3.4.8　例题 8 存储器芯片连接图（2）

9．若有一个具有 20 位地址的 32 位字长的存储器，则：

（1）计算存储器最多能存储多少字节的信息？

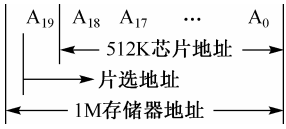
存储器单元数为 2^{20} = 1 M，每个存储单元存放一个 32 位的字，故最多可存储 1 M ×32 位 = 4 MB 的信息。

（2）用 512 K×8 位/片的 SRAM 芯片组成该存储器，需要多少芯片？

用 4 片 512 K×8 位/片的 SRAM 芯片进行位扩展，构成一个 512×32 位的存储模块，用两个这样的存储模块作字扩展，即可构成 1 M×32 位的存储空间。所以需要 4×2 = 8 片 512 K×8 位/片的芯片。

（3）若地址线为 A_{19}（高）～A_0（低），那么哪些作芯片地址用，哪些作芯片选择用？并写出片选信号逻辑式。

地址分配如下：



512 K×8 位芯片的芯片地址为 A_{18}～A_0，A_{19} 用于芯片选择，$\overline{A}_{19}$ 选择 1 M×32 位的前 512 K，即 $CS_0 = \overline{A}_{19}$，$CS_1 = A_{19}$。

（4）若 CPU 的地址总线为 A_{19}（高）～A_0（低），数据总线 D_{31}～D_0，读写控制信号为 R/$\overline{W}$，试设计出该存储器芯片级逻辑图。

其芯片连接图如图3.4.9所示。

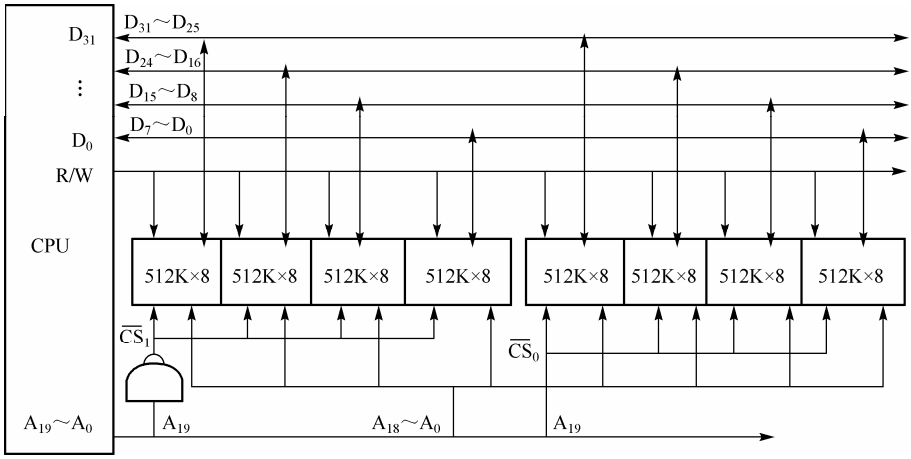


图 3.4.9　例题 9 存储器芯片连接图

10．某机 I/O 空间与主存统一编址，该机字长 16 位，存储空间（按字编址）分配如下：0000H～3FFFH 为 ROM 区，4000H～7BFFH 为 RAM 区，7C00H～7FFFH 为 I/O 地址空间。若选用（即必须用）4 KB/片的 RAM 芯片、8 KB/片的 EPROM 芯片和 4 K×16 位/片的 EPROM 芯片来组成存储器，该地址总线为 A_{15}～A_0（低），数据总线为 D_{15}～D_0。则：

（1）三种存储器芯片各需多少片？

根据题意，三种芯片都必须选用。

分析：ROM 区（0000H～3FFFH）存储容量为 16 K 字，故选用二片 8KB/片的 EPROM 芯片进行位扩展构成一个 8 K 字（即 8 K×16 位）的 ROM 模块；选用二片 4 K×16 位/片的 EPROM 芯片（分别为 ROM_1 和 ROM_2）进行字扩展构成 4 K 字的 ROM，于是用 ROM_0、ROM_1 和 ROM_2 进行字扩展则构成 16 K 字的 ROM 区。

RAM 区（4000H～7BFFH）存储容量为 15 K 字，且只能用 4 KB/片的 RAM 芯片。故用 4 KB/片 RAM 芯片两片进行位扩展，组成一个 4 K×16 位（即 4 K 字）的存储模块，需 4 个这种存储模块（分别为 RAM_0、RAM_1、RAM_2 和 RAM_3）进行字扩展，构成 16 K 字的 RAM 存储空间。

I/O 地址空间（7C00H～7FFFH）为 1K 字，位于 16 K 字的 RAM 区地址最高端。

根据上述分析，该存储器需：8 KB EPROM 芯片 2 片，4 K 字 EPROM 芯片 2 片，4 KB RAM 芯片 8 片。

（2）每块芯片需分配哪几位地址？

芯片地址

8 KB EPROM 芯片：A_{12}～A_0。

4 K（包括 4 K 字和 4 KB 芯片）芯片：A_{11}～A_0。

（3）应设几个片选信号？写出各片选信号的逻辑式，以实现对 ROM 区和 RAM 区的访问。

片选信号，将 ROM 和 RAM 区地址空间的十六进制表示转换为二进制表示，空间分配如图3.4.10 所示。

		$A_{15}\cdots A_0$						
0000H		0000	0000	0000	0000	8 KB EPROM	8 KB EPROM	ROM_0
		0001	1111	1111	1111			
		0010	0000	0000	0000	4 K×16 位 EPROM		ROM_1
		0010	1111	1111	1111			
		0011	0000	0000	0000	4 K×16 位 EPROM		ROM_2
3FFFH		0011	1111	1111	1111			
4000H		0100	0000	0000	0000	4 KB RAM	4 KB RAM	RAM_0
		0100	1111	1111	1111			
		0101	0000	0000	0000	4 KB RAM	4 KB RAM	RAM_1
		0101	1111	1111	1111			
		0110	0000	0000	0000	4 KB RAM	4 KB RAM	RAM_2
		0110	1111	1111	1111			
	2K	0111	0000	0000	0000	4 KB RAM	4 KB RAM	RAM_3
		0111	0111	1111	1111			
	1K	0111	1000	0000	0000			
7BFFH		0111	1011	1111	1111			
7C00H	1K	0111	1100	0000	0000			
7FFFH		0111	1111	1111	1111			

图 3.4.10　例题 10 存储空间分配图

分析：存储空间共 32 K 字，可分为 4 个 8 K 字，故用 $A_{14}A_{13}$ 译码输出来区分；每个 8 K 字存储空间可分为前后 4 K 字，用 A_{12} 来区分。于是可写出 ROM_0、ROM_1、ROM_2、RAM_0、RAM_1、RAM_2 的片选逻辑。在 RAM_3 的 4 K 字存储空间中，其高端 1 K 字存储空间 7C00H～7FFFH 是 I/O 地址空间，不作为主存空间用，故 RAM_3 的片选逻辑应扣除这 1 K 字的存储空间，于是全部片选信号逻辑式如图3.4.11 所示。

	$A_{15}A_{12}$			A_0			
0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	8 KB EPROM		系统程序区 8 KB EPROM
8191	0 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1			
8192	0 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K	$SRAM_0$ （16 KB）	工作区 SRAM （50 KB）
16383	0 0 1 1	1 1 1 1	1 1 1 1	1 1 1 1			
16384	0 1 0 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K		
24575	0 1 0 1	1 1 1 1	1 1 1 1	1 1 1 1			
24576	0 1 1 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K	$SRAM_1$ （16 KB）	
32767	0 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1			
32768	1 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K		
40959	1 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1			
40960	1 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K	$SRAM_2$	
49151	1 0 1 1	1 1 1 1	1 1 1 1	1 1 1 1			
49152	1 1 0 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K		
57343	1 1 0 1	1 1 1 1	1 1 1 1	1 1 1 1			
57344	1 1 1 0	0 0 0 0	0 0 0 0	0 0 0 0	8 K	$SRAM_3$	
59391	1 1 1 0	0 1 1 1	1 1 1 1	1 1 1 1			
59392	1 1 1 0	1 0 0 0	0 0 0 0	0 0 0 0	4 K		4 KB（空）
63487	1 1 1 1	0 1 1 1	1 1 1 1	1 1 1 1			
63488	1 1 1 1	1 0 0 0	0 0 0 0	0 0 0 0	2 K		I/O 地址空间 （2 KB）
65535	1 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1			

图 3.4.11　例题 11 存储空间分配图

$$
\begin{aligned}
&\mathrm{ROM_0} \quad CS_0 = \overline{A}_{15}\overline{A}_{14}\overline{A}_{13}\\
&\mathrm{ROM_1} \quad CS_1 = \overline{A}_{15}\overline{A}_{14}A_{13}\overline{A}_{12}\\
&\mathrm{ROM_2} \quad CS_2 = \overline{A}_{15}\overline{A}_{14}A_{13}A_{12}\\
&\mathrm{RAM_0} \quad CS_3 = \overline{A}_{15}A_{14}\overline{A}_{13}\overline{A}_{12}\\
&\mathrm{RAM_1} \quad CS_4 = \overline{A}_{15}A_{14}\overline{A}_{13}A_{12}\\
&\mathrm{RAM_2} \quad CS_5 = \overline{A}_{15}A_{14}A_{13}\overline{A}_{12}\\
&\mathrm{RAM_3} \quad CS_6 = \overline{A}_{15}A_{14}A_{13}A_{12}\overline{A_{11}A_{10}}
\end{aligned}
$$

11．某机 I/O 设备与主存统一编址，地址总线为 A_{15}～A_0（低），双向数据总线为 D_7～D_0，存储器请求信号为 $\overline{\mathrm{MREQ}}$（低电平允许访存），读/写控制信号为 $\mathrm{R}/\overline{\mathrm{W}}$。主存空间分配如下：0～8191 为系统程序区，由 EPROM 芯片组成；8192～59491 为工作区，由 RAM 芯片组成；最大地址的 2 K 地址空间为 I/O 地址空间，上述地址为十进制表示，按字节编址，现有如下芯片：

EPROM：8 K×8 位/片

SRAM：16 K×8 位/片，8 K×4 位/片，4 K×8 位/片，2 K×8 位/片，1 K×8 位/片。

请从上述芯片中恰当选择芯片设计该计算机的主存储器。

（1）说明选择哪些芯片？各需多少？

（2）画出主存储器逻辑图，注明各信号线，写出各片选信号逻辑式。

主存空间按 16 位地址计算，为 2^{16} = 64 K，所以允许主存最大容量为 64 KB，根据题意系统程序区范围 0～8191（十进制），相应的十六进制地址范围为 0000H～1FFFH，所以 EPROM 的容量应为 8 KB，工作区地址范围 8192～59491，相应的十六进制地址范围为 2000H～E7FFH，所以工作区（RAM 区）的容量应为 50 KB；I/O 地址空间（2 K）的地址范围为 F800H～FFFFH，这不需要存储芯片。

（1）芯片选择

系统存储区存储容量为 8 KB，选用 8 K×8 位/片的 EPROM 芯片 1 片。工作区（RAM 区）50 KB，可用不同的 SRAM 芯片来组合构成，这也有多种组成方式，例如：

① 16 K×8 位/片：3 片；2 K×8 位/片：1 片。

② 8 K×8 位/片：6 片，2 K×8 位/片：1 片。

③ 8 K×8 位/片：7 片。

④ 4 K×8 位/片：12 片，2 K×8 位/片：1 片。

⑤ 4 K×8 位/片：13 片。

第一种方案用的芯片最少，可选用此方案，所需芯片种类和数量如下：

EPROM：8 K×8 位/片，一片。

SRAM：16 K×8 位/片，三片；

2 K×8 位/片，一片。

（2）主存储器逻辑图

根据题意和上述分析与芯片选择，可得到如图 3.4.11 所示的存储空间分配。

（3）片选逻辑

整个存储空间分成 6 个可独立访问的区域：EPROM、$SRAM_0$、$SRAM_1$、$SRAM_2$、$SRAM_3$ 和 I/O 地址空间。64 K 个编址单元可看成 8 个 8 K 编址单元构成。EPROM 占一个 8K 编址单元，$SRAM_0$、$SRAM_1$、$SRAM_2$ 依次各占两个 8 K 编址单元，$SRAM_3$ 和 I/O 地址空间在同一个 8 K 编址单元中，但它们之间地址不连续，中间有 4 K 空区。因此，可以用 A_{15} A_{14} A_{13} 作译码器输入，以确定当前地址是访问哪一个 8 K 空间，进而再产生芯片的片选信号 $\overline{CS_i}$。

EPROM：8 KB，芯片（片内）地址线 A_{12}～A_0（低），共 13 根，片选逻辑为 $CS_0 = \overline{A}_{15}\overline{A}_{14}\overline{A}_{13}$。

$SRAM_0$：16 KB，芯片地址线 A_{13}～A_0（低），共 14 根。片选逻辑由 16K 的前 8K 和后 8K 一起构成，片选逻辑为 $CS_1 = \overline{A}_{15}\overline{A}_{14}A_{13} + \overline{A}_{15}A_{14}\overline{A}_{13} = \overline{A}_{15}$。

$SRAM_1$：16 KB；片选逻辑为 $CS_2 = \overline{A}_{15}A_{14}A_{13} + A_{15}\overline{A}_{14}\overline{A}_{13}$。

$SRAM_2$：16 KB；片选逻辑为 $CS_3 = A_{15}\overline{A}_{14}A_{13} + A_{15}A_{14}\overline{A}_{13}$。

$SRAM_3$：2 KB，位于 64 K 空间的最后 8 K 的最前 2 K 中，可用 $A_{12}A_{11}$ 的四种状态来区分 8 K 中的那个 2 K。芯片地址线为 A_{10}～A_0，共 11 根；片选逻辑为 $CS_4 = A_{15}A_{14}A_{13}\overline{A}_{12}\overline{A}_{11}$。

I/O 地址空间：2 KB，位于 64 K 空间的最后 8 K 的最后 2 K 中。片选逻辑为 $CS_5 = A_{15}$ A_{14} A_{13} A_{12} A_{11}。

存储器逻辑如图3.4.12所示。

3.4.2 第 4 章习题解答与解题思路

1．简要解释下列名词术语。

【答】

虚拟存储器：依靠操作系统的支持来实现的，为用户提供一个比实际内存大的可访问存储器空间，即在软件编程上可使用的存储器。

随机存储器 RAM：按给定地址随机地访问任一存储单元，访问时间与单元位置无关。

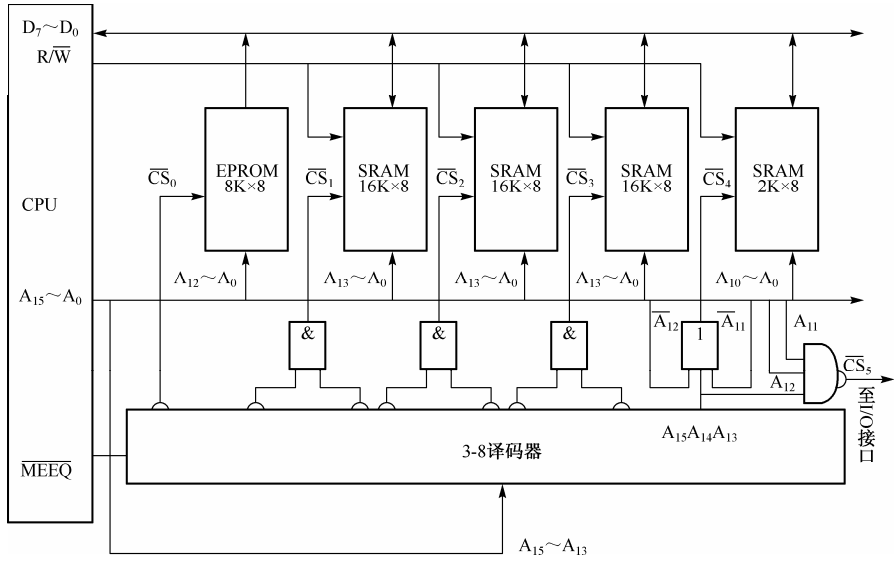


图 3.4.12　例题 11 存储器连接逻辑图

只读存储器 ROM：在正常工作中只能读出、不能写入的存储器。

存取周期：指存储器做连续访问操作过程中一次完整的存取操作所需的全部时间。

数据传输率：数据传入或传出存储器的速率。

动态刷新：对动态存储器中原存信息为 1 的电容补充电荷，称为动态刷新。

直接映像 Cache：将主存与 Cache 的存储空间划分为若干大小相同的页，每个主存页只能复制到某一个固定的 Cache 页中。

全相联映像 Cache：将主存与 Cache 的存储空间划分为若干大小相同的页，主存的每一页可以映像到 Cache 的任一页上。

组相联映像 Cache：将主存与 Cache 都分组，主存中一个组内的页数与 Cache 的分组数相同，每一组 Cache 中含有若干页（一般页数较少），则主存中的各页与 Cache 的固定组号有映像关系，可自由映像到对应的 Cache 组中任一页。

段页式虚拟存储器：将程序按其逻辑结构分段，每段再分为若干大小相同的页，主存空间也划分为若干同样大小的页。相应地，建立段表与页表，分两级查表实现虚实地址的转换。以页为单位调进或调出主存，按段共享与保护程序及数据。

相联存储器：一种按内容寻址的存储器，是根据所存信息的全部特征或部分特征进行存取的存储器，称为相联存储器。

2．某半导体存储器容量 8 K×8 位，可选 RAM 芯片容量为 2K×4/片。地址总线 A_{15}～A_0（低），双向数据线 D_7～D_0（低），由 R / $\overline{W}$ 线控制读写。请设计并画出该存储器逻辑图，并注明地址分配与片选逻辑式及片选信号极性。

【答】

设计存储器需要由总容量确定可选存储器芯片的数量，由于每片芯片容量通常低于总量，要用若干块芯片组成，这样需要在位数和单元数上进行扩展。其次，要考虑如何连接有关存储芯片的地址线、片选信号线、数据线和控制信号线。

存储器由若干存储芯片组成，每个芯片包含若干存储单元。地址分配的原则是将存储器的低位地址分配给存储芯片，以选择芯片内的存储单元；高位地址分配给片选逻辑，译码后产生片选信号，选择某个存储芯片。

（1）确定芯片数量：本题存储容量为 8 K×8 位，可选的芯片容量为 2 K×4/片。在构成存储器时，可选进行位扩展，由 2 块 2 K×4/片的芯片为一组，构成 2 K×8 的存储模块。由 4 组 2 K×8 的存储模块构成 8K×8 的存储器。

位扩展：　2K×4/片× 2 →　2 K×8

单元扩展：2K×8　×4 →　8 K×8

所以共需芯片数量为 2×4 = 8 片。

（2）地址分配和片选逻辑设计

本题存储容量为 8 KB 单元，占 16 位地址总线的低 13 位 A_{12}～A_0。用这 13 位地址可寻址整个 8 KB 存储空间，接下来再对这 13 位地址进行分配。

每块芯片容量为 2 K×4/片，需 11 位地址对片内单元寻址。组成 8 KB 存储器需 4 组芯

片，所以需用 4 个片选信号对它们进行选择。因此，将 13 位地址中的低 11 位地址 A_{10}～A_0 分配给各存储芯片，加至各芯片的地址端，剩下的高两位地址 A_{12}、A_{11} 送片选逻辑译码，产生 4 个片选信号 CS_0、CS_1、CS_2、CS_3，这 4 个片选信号的逻辑式为：

$$CS_0 = \overline{A}_{12}\overline{A}_{11}$$

$$CS_1 = \overline{A}_{12}A_{11}$$

$$CS_2 = A_{12}\overline{A}_{11}$$

$$CS_3 = A_{12}A_{11}$$

（3）画连接图，需要连接的信号线包括地址线、数据线、控制线和片选信号线，其连接方法为：

芯片地址线连接至：A_{10}～A_0。

数据线：一组连接至高 4 位 D_7～D_4，一组连接至低 4 位 D_3～D_0。

片选信号分开连，控制信号线 $R/\overline{W}$ 连接至每块芯片，其具体连接图见图 3.4.13。

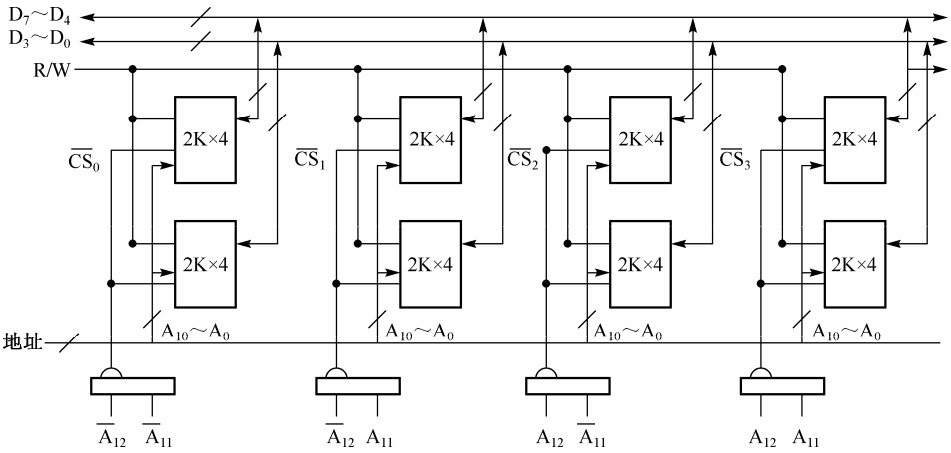


图 3.4.13　存储器逻辑图

3．某半导体存储器容量 7 K×8 位。其中固化区 4 K×8，可选 EPROM 芯片 2 K×8/片。随机读写区 3 K×8，可选 SRAM 芯片 2 K×4/片、1 K×4/片。地址总线 A_{15}～A_0（低），双向数据总线 D_7～D_0（低），$R/\overline{W}$ 控制读写。另有控制信号 $\overline{MREQ}$，低电平时允许存储器工作。请设计并画出该存储器逻辑图，并注明地址分配与片选逻辑式及片选信号极性。

【答】

（1）芯片选择：需 2 块 2 K×8/片的 EPROM 芯片进行单元扩展，构成 4 K×8 的固化区；2 块 2 K×4/片及 2 块 1 K×4/片的 SRAM 芯片进行位扩展和单元拼接构成 3 K×8 的随机读写区。

（2）地址分配及片选逻辑设计：7 K 字节单元占 16 位地址线的低 13 位 A_{12}～A_0，连接至各芯片的地址分别为：

2 K×8 的 EPROM 占 11 位地址线：A_{10}～A_0。

2 K×4 的 SRAM 占 11 位地址线：A_{10}～A_0。

1 K×4 的 SRAM 占 10 位址线：A_9～A_0。

组成 7 K 的存储器共有 4 组芯片，2 组 2 K×8 的 EPROM、1 组 2 块 2 K×4 的 SRAM、1 组 2 块 1 K×4 的 SRAM 芯片，2 K 单元的芯片用剩下的高两位地址 A_{12}、A_{11} 送片选逻辑译码，产生片选信号 CS_0、CS_1、CS_2，1 K 单元的芯片用剩下的高三位地址 A_{12}、A_{11}、A_{10} 送片选逻辑译码，产生片选信号 CS_3，这 4 个片选信号的逻辑式为：

$$CS_0 = \overline{A}_{12}\overline{A}_{11}$$

$$CS_1 = \overline{A}_{12}A_{11}$$

$$CS_2 = A_{12}\overline{A}_{11}$$

$$CS_3 = A_{12}A_{11}\overline{A}_{10}$$

该存储器逻辑图见图3.4.14。

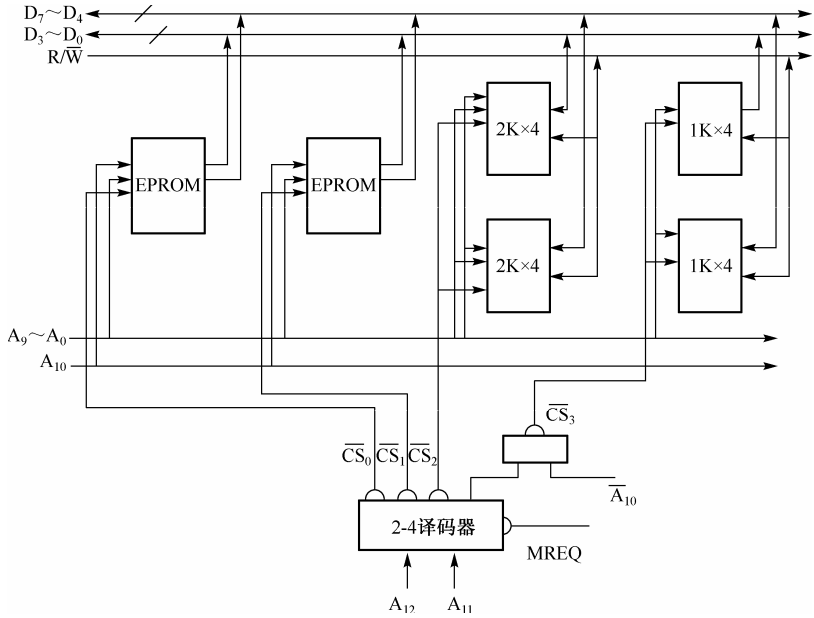


图 3.4.14　存储器逻辑图

4．某机地址总线 16 位 A_{15}～A_0（低），访存空间为 64 KB。外围设备与主存统一编址，即将外围设备接口中有关寄存器与主存单元统一编址，I/O 空间占用 FC00H～FFFFH。现用 2164 构成主存储器，请设计并画出该存储器逻辑图，并画明芯片地址线与总线的连接逻辑，以及行选信号与列选信号的逻辑式，使访问外设时不访问主存。

【答】

由 I/O 所占用的地址空间为 FC00～FFFF，即

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	…	A_0
1	1	1	1	1	1	0	…	0
								M
1	1	1	1	1	1	1	…	1

可知，I/O 占用了 10 位地址空间，所以 I/O 的单元数为 1KB，因此主存的地址空间为 63 KB。2164 为 64 K×1 位容量的芯片，因此需 8 块 2164 芯片构成主存储器。

每片 2164 提供一位数据送数据总线，构成 8 位数据，64K 单元需要 16 位地址线寻址，

由于 2164 只有 8 位地址线，所以采用地址的分时复用技术。16 位地址分两次送入芯片，先送入行地址，再送入列地址，相应地需要行选择信号 $\overline{\text{RAS}}$ 和列选择信号 $\overline{\text{CAS}}$ 来标明当前送入的是行地址还是列地址。

由于 I/O 与主存空间统一编址，占用 64 K 地址空间的最后 1 K 地址空间，因此当高 6 位地址为全 1 时，应选中 I/O 地址空间，否则访问主存空间，而行选信号和列选信号的逻辑为 $\text{RAS}=\text{CAS}=A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}$。

该存储器逻辑图见图3.4.15。

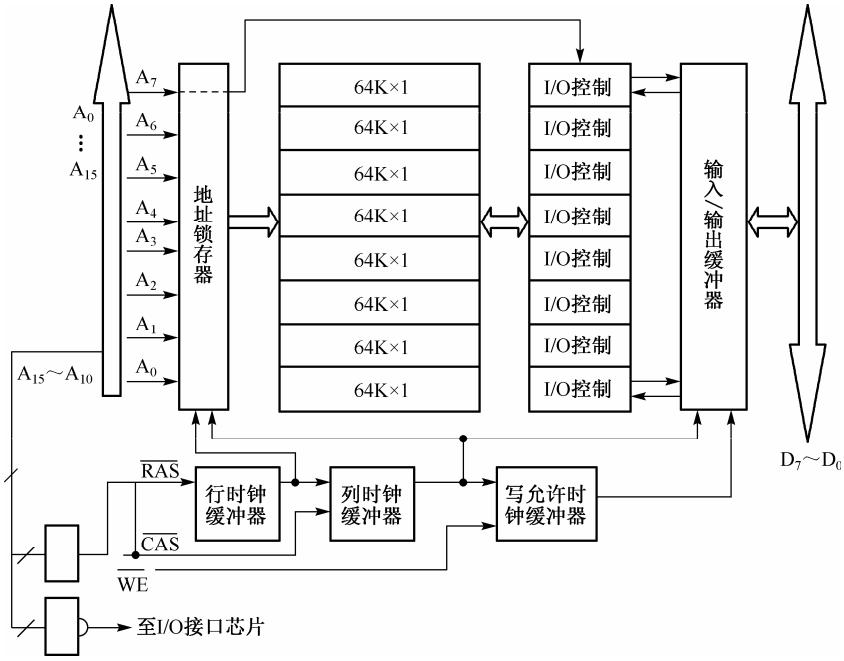


图 3.4.15　存储器逻辑图

5．某半导体存储器容量为 14 KB，其中 0000H～1FFFFH 为 ROM 区，2000H～37FFH 为 RAM 区，地址总线 A_{15}～A_0（低），双向数据总线 D_7～D_0（低），读写控制线 $\text{R}/\overline{\text{W}}$。可选用的存储芯片有 EPROM（4 KB /片）和 RAM（2 K×4 /片）。

（1）计算所需各类芯片的数量。

（2）说明加到各芯片的地址范围值和地址线。

（3）写出各片选信号的逻辑式。

（4）画出该存储芯片级逻辑图，包括地址总线、数据线、片选信号线（低电平有效）及读写信号线的连接。

【答】

（1）芯片容量，ROM 区 8 K，RAM 区 6 K，因此 EPROM 需要 2 片，RAM 需要 6 片。

（2）各芯片的地址范围：

ROM：0000H~0FFFH，1000H~1FFFH

RAM：2000H~27FFH，2800H~2FFFH，3000H~37FFH

ROM 地址线为 A_{11}~A_0，RAM 地址线为 A_{10}～A_0。

（3）片选信号的逻辑表达式

$\overline{CS}_0=\overline{A}_{13}\overline{A}_{12}$，$\overline{CS}_1=\overline{A}_{13}A_{12}$，$\overline{CS}_2=A_{13}\overline{A}_{12}\overline{A}_{11}$，

$\overline{CS}_3=A_{13}\overline{A}_{12}A_{11}$，$\overline{CS}_4=A_{13}A_{12}\overline{A}_{11}$，$A_{15}A_{14}$ 为 00。

（4）逻辑结构图如图 3.4.16 所示。

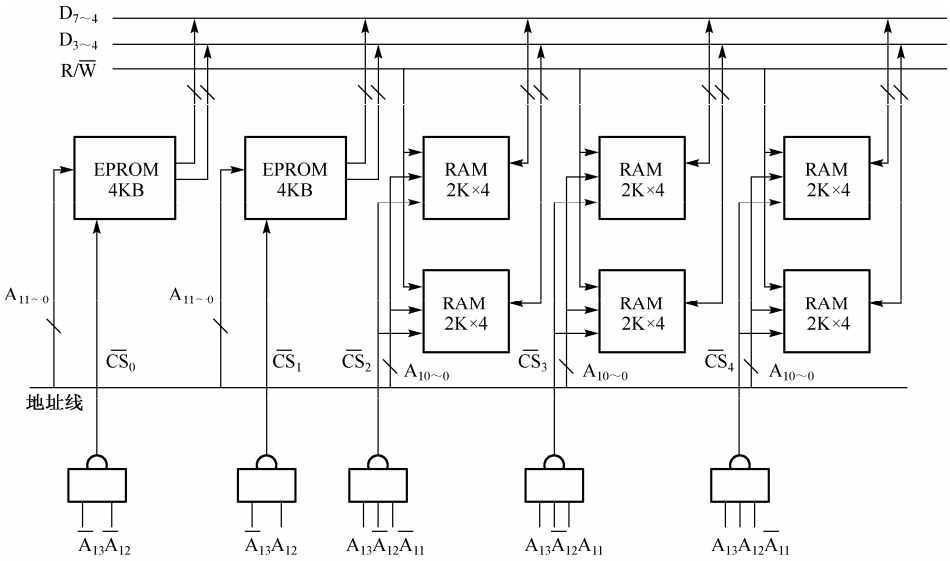


图 3.4.16　逻辑结构图

6．SRAM 和 DRAM 各依靠什么原理存储信息？

【答】

SRAM 即静态半导体存储器，静态半导体存储器中的一位存储单元，实际上是一个双稳态触发器，靠交叉反馈存储信息。

DRAM 即动态半导体存储器则是靠电容电荷存储信息，如电容上有电荷代表存放的信号 1，电容无电荷为存放的信号 0。

7．设某机主存 1 MB，用 1 MB/片的 DRAM 芯片构成，芯片最大刷新周期 2 ms。问在 2 ms 之内至少应该安排几个刷新周期？

【答】

由于动态存储器的刷新是按行进行的，每一行中的位单元同时进行刷新，所以刷新不是按单元地址进行的，而是按行地址进行的。存储芯片阵列的行数是多少就需要在最大刷新周期中安排多少个刷新周期。

1 MB/片的 DRAM 芯片存储器若为 1024×1024 矩阵，那么在 2 ms 之内至少应该安排 1024 个刷新周期。

8．动态刷新周期的安排方式有哪几种？简述它们的安排方法，并指出在下列情况中可

选取哪一种或几种刷新方式。

a）教学用单板计算机。

b）常用个人计算机。

c）带多个分时终端的超小型计算机。

d）如果主存的存取周期 200 ns，CPU 平均访存时间约占主存工作时间的 90%。

e）主存的存取周期 200 ns，CPU 平均访存时间约占主存工作时间的 40%。

【答】

动态刷新周期的安排方式有以下 3 种：

① 集中刷新方式：在要求的时间间隔内集中安排所有刷新周期，其余时间用于正常访存工作。

② 分散刷新方式：将各刷新周期分散的安排在各存取周期之后，即将每个存取周期分为两部分，前半期用于正常读/写或保持，后半期用于刷新。

③ 异步刷新方式：按芯片阵列的行数决定所需要的刷新周期数，并将各刷新周期分散安排在刷新要求的时间间隔周期中，每隔固定时间提出一次刷新请求，安排一个刷新周期。

题目中所述各种情况可选择的刷新方式如下。

① 教学用单板计算机：由于该计算机是为了教学用，主要让学生理解刷新的基本原理和实现方法，所以该类计算机可选择三种刷新方式中的任何一种。

② 常用个人计算机：采用异步刷新，由于集中刷新会有明显的死区，而分散刷新只适用于低速的系统，都不适合于个人计算机，异步刷新是大多数计算机系统采用的方式。

③ 带多个分时终端的超小型计算机：可采用分散刷新方式，可将刷新安排在各分时终端的时间片中。

④ 如果主存的存取周期 200 ns，CPU 平均访存时间约占主存工作时间的 90%：采用异步刷新。由于 CPU 平均访存时间要占主存工作时间的 90%，集中刷新死区过长，而分散刷新的次数太多，占用主存的时间太长，只有异步刷新方式对主存工作的时间占用最少。

⑤ 主存的存取周期 200 ns，CPU 平均访存时间约占主存工作时间的 40%：可采用集中刷新的方式。由于 CPU 平均访存时间只占主存工作时间的 40%，可在剩余的时间内安排集中刷新，这样可使刷新电路简单易于设计。

9．若对磁表面存储器写入代码 10011，请画出 NRZ-1 制、PE 制、FM 制、M^2F 制等记录方式的写电流波形。

【答】

（1）NRZ-1 制的写入规律为：写 0 时，写入电流维持原方向不变，写 1 时，写入电流方向翻转。其写电流波形如图3.4.17 所示。

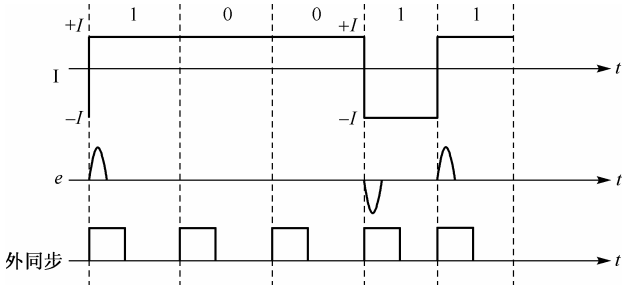


图 3.4.17　NRZ-1 制写电流波形

（2）调相制的写入规律为：写 0，在位单元中间位置让写入电流负跳变，由+*I*→–*I*；写 1，在位单元中间位置让写入电流正跳变，由–*I*→+*I*。其波形如图3.4.18 所示。

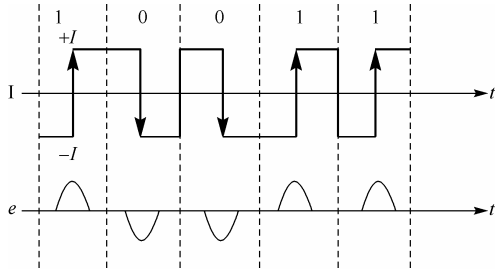


图 3.4.18　调相制写电流波形

（3）FM 制的写入规律为：在位单元中间记录数据信息。如果写入 0，则位单元中间不变。如果写入 1，则写入电流在位单元中间改变一次方向。其波形如图3.4.19 所示。

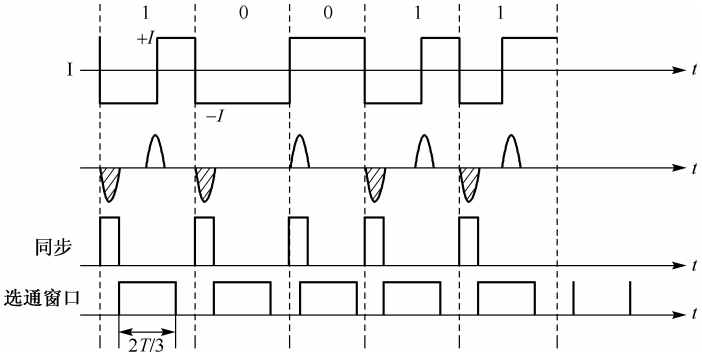


图 3.4.19　FM 制写电流波形

（4）M^2F 改进型调频制的写入规律为：写 1 时，在位单元中间改变写入电流方向；写入两个以上 0 时，在它们的交界处改变写入电流方向。其波形如图3.4.20 所示。

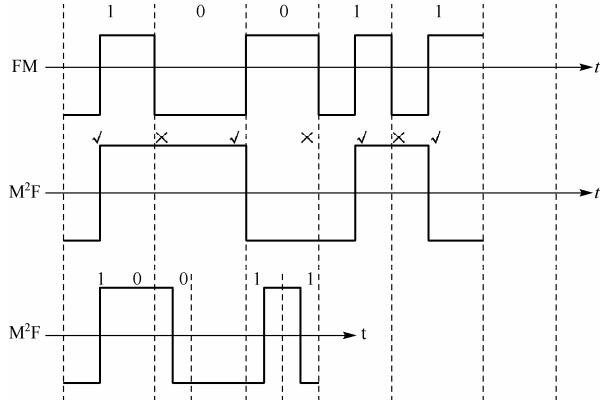


图 3.4.20　M^2F 制写电流波形

10．欲写入代码 10011100（不连校验位）：

（1）采用奇校验，写出配校验位后的校验码；

（2）采用偶校验，写出配校验位后的校验码。

【答】

奇校验的规律为：使整个校验码中“1”的个数为奇数个，偶校验的规律为：使整个校验码中“1”的个数为偶数个。因此配校验位后，分别采用奇、偶校验的校验码为：

	有效信息	校验位
奇校验	10011100	1
偶校验	10011100	0

11．欲写入 8 位有效信息 01101101，试将它编为海明校验码。以表格形式说明其编码方法，并分析所选用的编码方案具有什么样的检错与纠错能力。

【答】

（1）确定信息分为几组，增设几位校验位。有效信息为 $A_1A_2A_3A_4A_5A_6A_7A_8=01101101$，其位数 K=8，设分为 r 组，每组增设一个校验位，因此共有 r 位校验位，校验位与有校信息组成 n 位的海明校验码。

校验时每组产生一位校验信息，组成一个 r 位的指误字，可指出 2^r 种状态，其中全 0 表示无错，余下的组合可分别指明（2^r-1）位中的某一位错误。因此 r 的值应满足：$n=K+r\leqslant 2^r-1$ 的要求，所以 $r=4$，信息应分为 4 组。

（2）如何分组：等编有效信息为 01101101，增设校验位 P_1、P_2、P_3、P_4，分为 4 组，可产生 4 位指误字 G_1、G_2、G_3、G_4。编码方案为证指误字代码与出错位序号相同，如各组采用偶校验。各位的排列和分组方案如表 3-4-1 所示。

表 3-4-1　海明校验码的分组方案

	1	2	3	4	5	6	7	8	9	10	11	12	指误字
	P_1	P_2	A_1	P_3	A_2	A_3	A_4	P_4	A_5	A_6	A_7	A_8	
第 4 组								√	√	√	√	√	G_4
第 3 组				√	√	√	√					√	G_3
第 2 组		√	√			√	√			√	√		G_2
第 1 组	√		√		√		√		√		√		G_1

（3）编码

第一组	第二组	第三组	第四组
$A_1A_2A_4A_5A_7 = 01010$	$A_1A_3A_4A_6A_7 = 01010$	$A_2A_3A_4A_8 = 1101$	$A_5A_6A_7A_8 = 1101$
$P_1 = 0$	$P_2 = 0$	$P_3 = 1$	$P_4 = 1$

最后得到的 12 位海明校验码为：000111011101，如表 3-4-2 所示。

表 3-4-2　海明校验码的检错、纠错模式

1	2	3	4	5	6	7	8	9	10	11	12	指误字
P_1	P_2	A_1	P_3	A_2	A_3	A_4	P_4	A_5	A_6	A_7	A_8	$G_4\ G_3\ G_2\ G_1$
												0 0 0 0
错												0 0 0 1
	错											0 0 1 0
		错										0 0 1 1
			错									0 1 0 0
				错								0 1 0 1
					错							0 1 1 0
						错						0 1 1 1
							错					1 0 0 0
								错				1 0 0 1
									错			1 0 1 0
										错		1 0 1 1
											错	1 1 0 0

12．将 4 位有效信息 1001 编成循环校验码，选择生成多项式 $X_3+X_1+X_0$。写出编码过程。

【答】

（1）编码方法

$M(x) = x^3+x^0$，　即 1001　　$(k = 4)$

$M(x)\cdot x^r = x^6+x^3$，　即 1001000　　$(r=3)$

$G(x)=x^3+x^1+x^0$，　即 1011　　$(r+1=4)$

$$\frac{M(x)}{G(x)} = \frac{1001000}{1011} = 1010 + \frac{110}{1011}$$　（模 2 除）

（2）编码后的校验码为：$M(x)\cdot x^3+R(x)=1001000+110=1001110$（模 2 加）。

13．双端口存储器与两个独立存储器有何不同？

【答】

双端口存储器和两个独立的存储器虽然都具有两个彼此独立的读/写口，每个读/写口都有一套独立的地址寄存器和译码电路，可以并行地独立工作，并且两个读/写口可以按各自接收的地址，同时读出或写入，或一个写入而另一个读出。但双端口存储器与两个独立的存储器不同的是，两套读/写口的访存空间相同，可以访问同一区间、同一单元；而两个独立的存储器的访存空间也是彼此独立的。

14．何谓单体多字并行主存系统？何谓多体交叉存取并行主存系统？

【答】

单体多字并行主存系统即是多个并行存储器共用一套地址寄存器，按同一地址码并行地

访问各自的对应单元。如假设有 n 个存储器顺序排列的 n 个字，每个字有 w 位，送入这 n 个存储器的地址均为 A，则这 n 个存储器同时访问各自的 A 号单元。我们也可以将这 n 个存储器视作一个大存储器，每个编址对应于 n 字×w 位，因而称为单体多字方式。

多体交叉并行主存系统即使用 n 个容量相同的存储器，或称为 n 个存储体，它们具有自己的地址寄存器、数据线、时序，可以独立编址地同时工作。各存储体的编址大多采用交叉编址方式，即将一套统一的编址，按序号交叉地分配给各个存储体。一段连续的程序或数据，将交叉地存放在几个存储体中，所以称为多体交叉存取并行主存系统。

15．某机主存容量 64 KB，用 2164DRAM 芯片构成。地址线 A_{15}～A_0（低），双向数据线 D_7～D_0（低），R/W 控制读写操作。请设计并画出该存储器逻辑图，对地址的行、列转换与数据线连接，应有明确描述。

【答】

2164 为 64 K×1 位的存储芯片，要构成 64 KB 容量的存储器，需要用 8 片 2164 做位拼接，每片提供一位数据，分别连接至双向数据线 D_7～D_0 上。

该存储器逻辑图见图3.4.21。

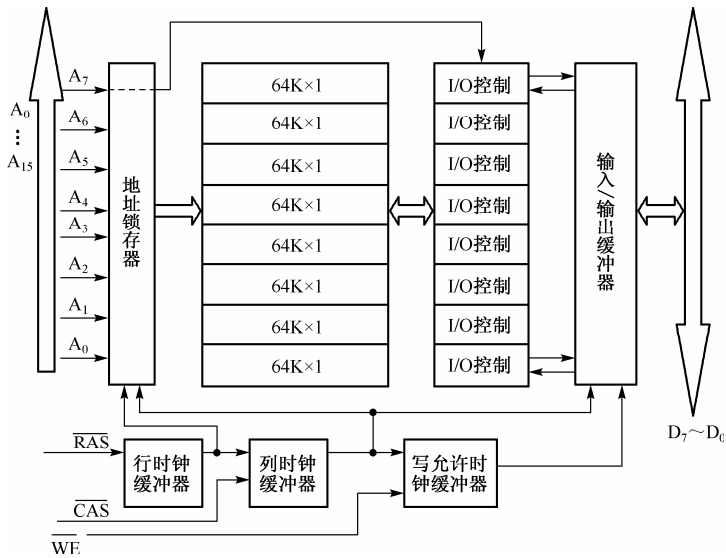


图 3.4.21　存储器连接逻辑图

16．某机主存容量 64 KB，用 2164DRAM 芯片构成。请为此设计一种动态刷新逻辑。

【答】

2164 为 64 K×1 位的芯片，芯片内部构成 4 个 128×128 的矩阵，由动态存储器刷新按行安排可知，该动态存储器在 2 ms 的时间间隔内需安排 128 个刷新周期。如以异步方式刷新，则每个刷新周期的时间间隔为：2 ms/128 = 15.625 μs。

异步刷新周期由一个单稳和一个非门组成的振荡器产生。该振荡器每隔 15.625 μs 产生一个脉冲信号，定时触发刷新请求触发器，于是每隔 15.625 μs 就产生一个刷新请求信号，其电路如图3.4.22 所示。该刷新请求信号触发器行计数器输出行地址，将该行所有位单元刷新。

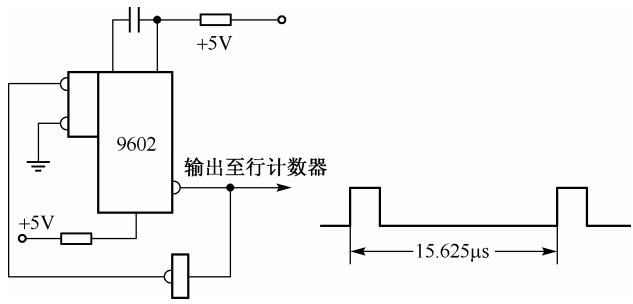


图 3.4.22　动态刷新逻辑

17．某海明编码 K=4，r=3，请为此设计其编码、译码、纠错逻辑。

【答】

该海明校验码分组规则如表 3-4-3 所示，其编码逻辑如图3.4.23所示。

表 3-4-3　海明校验码分组规则

	1	2	3	4	5	6	7	指误字
	P_1 A_4	P_2		A_1	P_3	A_2	A_3	
第 3 组				√	√	√	√	G_3
第 2 组		√	√			√	√	G_2
第 1 组	√		√		√		√	G_1

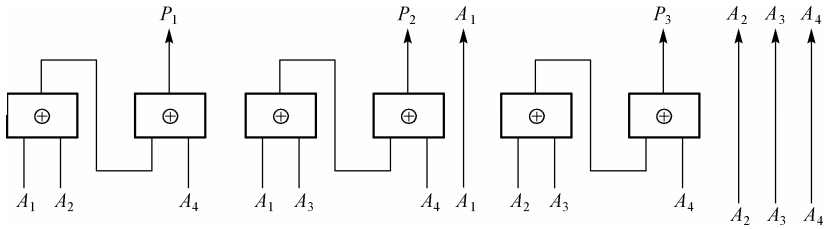


图 3.4.23　海明校验编码逻辑

其译码、纠错逻辑如图3.4.24所示，当从外部接收到一组海明码后，送往图3.4.22所示的译码电路进行分组奇偶检测，得到一组检错信息，译码后作为控制信号，决定是否要纠正。如 $G_3G_2G_1=0$，则最上面的一异或门控制端均为 0，输出 $P_1' \sim A_4' = P_1 \sim A_4$，即不需要纠正。若 $G_3G_2G_1=101$，则第 5 位异或门控制端为 1，$A_2' = A_2$，即将出错的第 5 位进行变反纠正，其余未错的各位保持不变。

18．某循环校验码，生成多项式 $X^3+X^1+X^0$ 请为此设计一个模 2 除法器。

【答】

模 2 除的分解步骤可归纳为两种类型：如果部分余数（包括被除数）最高位为 0，则将部分余数左移一位；　如果部分余数最高位为 1，则对生成多项式 $G(x)$作模 2 减，然后余数左移一位。由此，我们可以描述出模 2 除法器逻辑的主要组成，如图3.4.25所示。设置一个移位寄存器，模 2 减的特点是“减 0 不变、1 变反”，可用异或门实现；又因为每做一步除法都要左移一位，所以异或门的输出可直接送往高一位触发器的 D 端，即模 2 减与左移在一

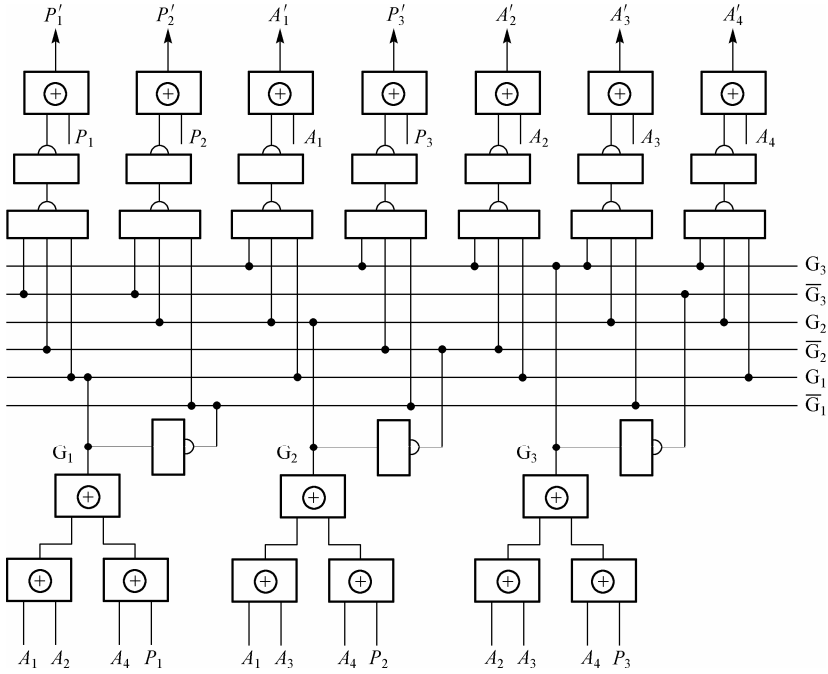


图 3.4.24　海明译码、纠错逻辑

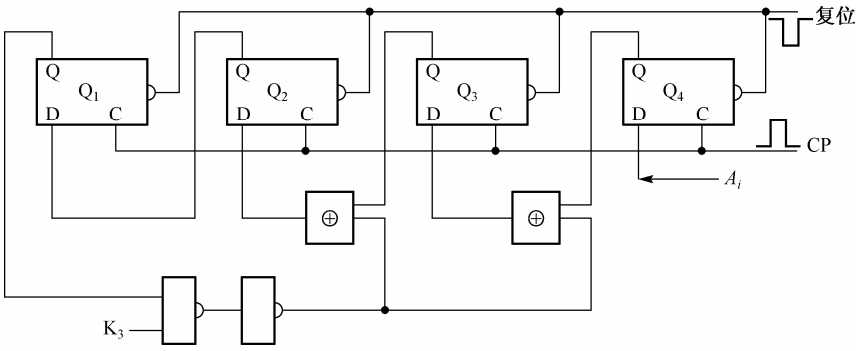


图 3.4.25　模 2 除法器

拍内完成。用最高位的状态去控制作模 2 减并左移，或单纯左移，控制各位模 2 减的条件就是生成多就是生成多项式 $G(x)$。当 $G(x)$确定后，如本题 $G(x)=1011$，可得到下列逻辑式：

$$D_1=(Q_1\cdot 0)\oplus Q_2=Q_2$$
$$D_2=(Q_1\cdot 1)\oplus Q_3=Q_1\oplus Q_3$$
$$D_3=(Q_1\cdot 1)\oplus Q_4=Q_1\oplus Q_4$$

以上逻辑式的含义是：当最高位 $Q_1=0$ 时有 $D_i=Q_{i+1}$ 即单纯左移；当 $Q_1=1$ 时，$Q_2Q_3Q_4$ 分别与 $G(x)$按位异或，再送往高位 D，即做模 2 减后左移一位。设置控制电位 K_3 控制模 2 除。D_4 条件要联系到整个编码、译码电路如何设计，应能串行输入待编信息或待译信息 A，也能在模 2 除过程中补 0。

19．某机主存按字节编址，而系统总线数据通路宽 32 位。请提出一种连接方案，以粗框描述，并简要说明。

【答】

如果系统总线数据通路宽 32 位，主存是按字节编址，则存储器系统可由 4 个存储体组成，存储体选择通过选择信号 $\overline{BE_3}$ ～ $\overline{BE_0}$ 实现，该选择信号可以通过用低们地址线译码得到。如果要传送一个 32 位数，那么 4 个存储体都被选中；若要传送一个 16 位数，则有 2 个存储体被选中；若传送的是 8 位数，则只有一个存储体被选中。

选择存储体与地址译码真的关系如表 3-4-4 所示。存储器与系统总线连接粗框图如图3.4.26所示。

表 3-4-4　存储体选择译码表

$\overline{BE_3}$	$\overline{BE_2}$	$\overline{BE_1}$	$\overline{BE_0}$	A_1	A_0	M_1	M_2	M_3	M_4	描述
0	0	0	0	0	0	1	1	1	1	访问 32 位数
1	1	0	0	0	0	1	1	0	0	访问 16 位数
0	0	1	1	0	1	0	0	1	1	访问 16 位数
1	1	1	0	0	0	1	0	0	0	访问 8 位数
1	1	0	1	0	1	0	1	0	0	访问 8 位数
1	0	1	1	1	0	0	0	1	0	访问 8 位数
0	1	1	1	1	1	0	0	0	1	访问 8 位数

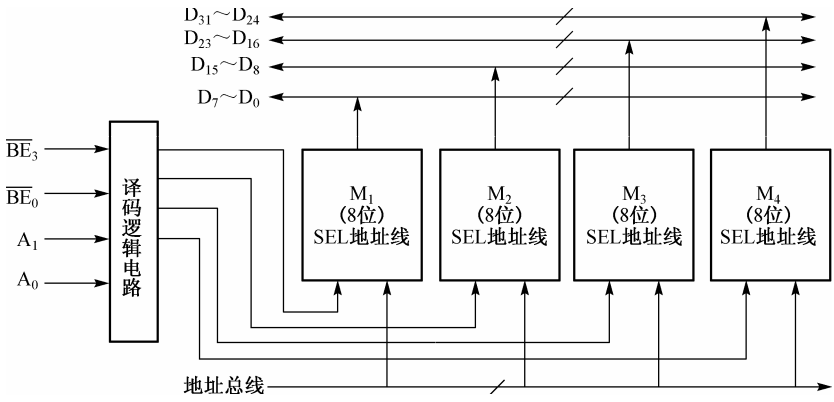


图 3.4.26　存储体连接粗框图

第二种方案则可以采用多体交叉存储器的连接方式，系统应由 4 个独立的存储体组成，各存储体容量相同，以字节为单位编址，每条存储体连接 8 位的数据线，存储体采用交叉编址方式，将主存的整个地址空间，按序号交叉地分配给 4 个存储体。其连接的粗框图如图3.4.27所示。

当需要传送一个 32 位数时，4 个存储体都被选中，当需要传送一个 16 位数据时，有 2 个存储体被选中；当需要传送 8 位数据时，则只有一个存储体被选中。

20．如果要求半导体存储器在完成读/写后产生信号 READY，并通过系统总线通知其他设备。请设计该信号的产生逻辑。

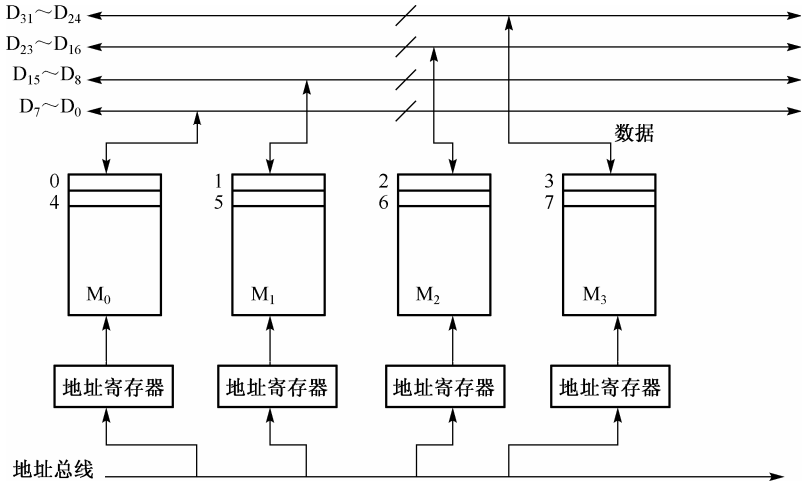


图 3.4.27　多体交叉存储器与系统总线连接方式

【答】

如果半导体存储器的读/写时间大于 CPU 所要求的时间，在这种情况下，为了保证 CPU 与存储器的时序的正确配合，就要产生 READY 信号，使 CPU 插入一个等待状态，并通过系统总线通知其他设备。能插入一个等待状态的 READY 信号电路逻辑如图3.4.28所示。

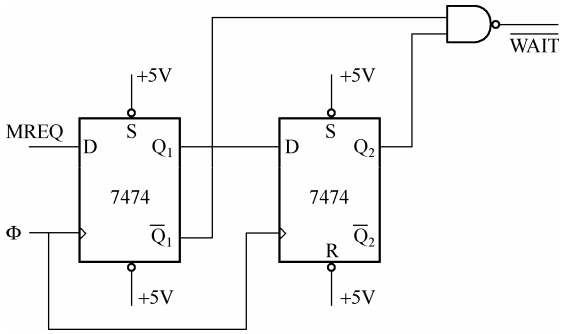


图 3.4.28　产生等待状态 READY 信号的逻辑

21．如果两台 CPU 通过各自的地址总线与数据总线共享一个半导体存储器，请为此设计存储器逻辑，使两组地址/数据线之间能够分离，且能处理访存冲突。

【答】

如果要两台 CPU 通过各自的地址总线与数据总线共享一个半导体存储器，可采用双端口存储器。双端口存储器有两套独立的地址寄存器、地址译码器、数据寄存器和读写电路，可同时接受来自两方面的访存请求，两组地址/数据线之间就能够分离。双端口存储器逻辑如图3.4.29所示。

以双端口存储器的方式，两个访问端口独立工作互不干扰，只有当两个端口试图在同一时间内访问同一地址单元时，才会发生冲突。这时根据两端口访问请求到达存储器的时间差，可由存储器的仲裁逻辑来决定首先为哪个 CPU 服务。

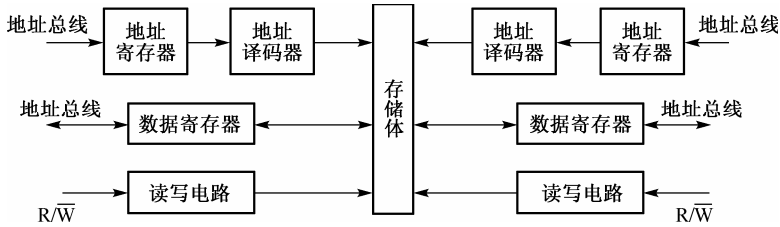


图 3.4.29　双 CPU 共享存储器连接图

22．如果需用常规存储芯片构成双端口存储器，允许存取周期延长。请为此设计存储器逻辑。

【答】

如果需用常规存储芯片构成双端口存储器，由于常规存储芯片没有两个独立的端口，所以需要为其设计一套控制逻辑，来实现双端口的功能，在接口中需要实现以下 4 方面的逻辑：两个数据缓冲，地址寄存，存储访问的控制逻辑，共享仲裁控制逻辑。

其存储器的逻辑及与系统的连接逻辑如图3.4.30 所示。

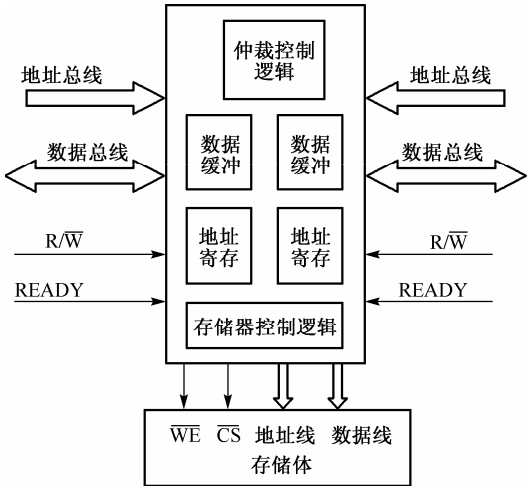


图 3.4.30　以常规存储芯片构成双端口存储器逻辑

如将以上存储系统用于双 CPU 共享的应用中，当两个 CPU 同时对存储器发出访问需求时，控制接口中的共享仲裁控制逻辑就会以访问请求的顺序，在第一个存储器访问周期时允许第一个 CPU 对存储器进行访问，同时向第二个 CPU 发出 READY 等待信号，然后在下一个存储器访问周期撤销该等待信号，允许第 2 个 CPU 对存储器进行访问。

23．设计一套硬件逻辑，以实现循环校验码的编码、译码、校正。画出粗框图。

【答】

假设待编码字为 $M(x)$共 k 位，生成多项式为 $G(x)= x^r + a_{r-1}x^{r-1} + \cdots + a_2x^2 + a_1x^1 + 1$。余数为 $R(x)$，则其编码电路如图3.4.31 所示。

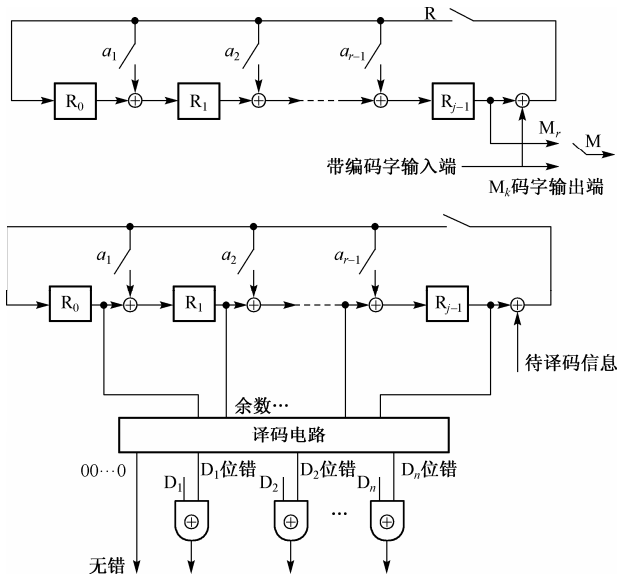


图 3.4.31　循环校验码的编码、译码、校正电路

当 a_i 为 1 时，开关 a_i 闭合，当 a_i 为 0 时，开关 a_i 断开。开始编码时，开关 M 置于 M_k 位置，同时开关 R 闭合，待编信息位由信息输入端从高位到低位串行送入，一方面输入信息从码字输出端输出，另一方面输入信息进入该编码电路做循环运算。当信息送入完毕时，寄存器 R_0、R_1、…、R_{j-1} 中的内容就是余数的值，这时断开开关 R，将开关 M 的位置设置至 M_r 处，则余数就拼接在待编信息位之后，实现编码过程。

24．设计一套子程序，以实现循环校验码的编码、译码、校正。画出流程图。

【答】

流程图如图3.4.32所示。

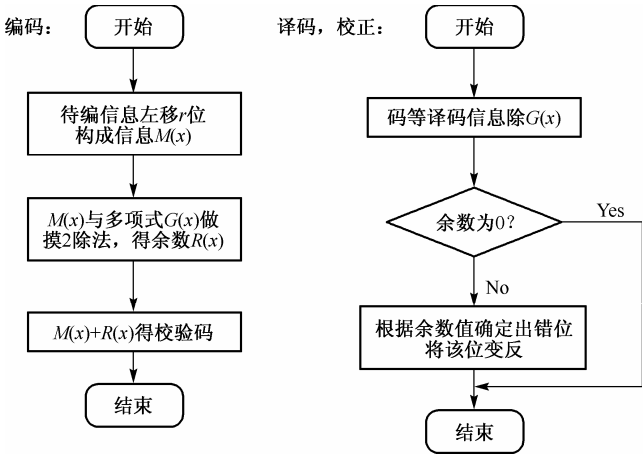


图 3.4.32　循环校验码的编码、译码、校正子程序流程

25．何谓磁盘的格式化容量和非格化容量？列出其计算公式。

【答】

格式化容量是指磁盘格式化以后，所有扇区内数据块的容量总和为：

字节数/扇区×扇区数/道×道数/面×面数

非格式化容量则是指磁盘的标称容量为：

内圈位密度×内圈周长×道数/面×面数

26．试比较当前光盘与磁盘两者的记录密度、平均存取时间和数据传输率三项性能指标。

【答】

具体见表 3-4-5。

表 3-4-5　光盘与磁盘的性能指标比较

	记录密度	平均存取时间	数据传输率
光盘	可达 300 Mbpi	>100 ms	DVD 可达 21.728 KBps
磁盘	可达 50 Tbpi	<15 ms	SATA 可达 600 MBps

27．主存和 Cache 之间的映像方式有哪几种？请简述每一种方式的基本思想，并分析每一种方式中如何通过变换内存地址以得到 Cache 的目标地址。

【答】

① 直接映射：每个主存页面只能映射到缓存中的固定页面位置。

② 全相联映射：每个主存页面均能映射到缓存中的任意页面位置。

③ 组相联映射：介于直接映射和全相联映射之间的一种映射方式,组间采用直接映射，组内采用全相联映射。

28．何谓随机存取？何谓顺序存取？何谓直接存取？请各举一例进行说明。

【答】

随机存取：指能按地址访问任意单元，访问的时间与地址无关，如内存。

顺序存取：访问时读/写部件按顺序查找目标地址，访问时间与数据位置有关，如磁带存储器。

直接存取：访问时读/写部件先直接指向一个小区域，再在该区域内顺序查找，访问时间与数据位置有关，如硬盘。

3.5　第 5 章习题解答及解题思路

3.5.1　第 5 章重点和难点解析

在输入/输出系统中，主要包含两大部件：总线和接口。它们起到将系统的部件有机地连接在一起并交换信息的作用。总线的设计要考虑的因素很多，但在各种因素中，总线的时序控制方式在很大程度上决定了总线与其他部件的连接方式、信息交换方式和控制方式。接口则要根据外设与主机的信息交换方式决定其设计的要素。以下通过实例说明总线、中断方式和接口、DMA 方式和接口设计的重点和难点。

1．试说明同步总线和异步总线的特点，比较各自的优缺点及应用场合。

同步总线的特点：有统一的时序划分，各项操作之间、各部件间的衔接有严格的时序控制，操作时间固定。

同步总线控制方式的优点：时序关系简单，时序划分规整，易于设计实现；其缺点是时间安排不够合理。

同步控制总线的应用场合主要为：CPU 内部、部件内部、系统总线或工作速度差异不大、传送距离近、传送时间固定的设备之间。

异步总线的特点为：没有统一的时序划分，各项操作之间、部件之间的衔接采用异步应答方式。

异步总线控制方式的优点为：时间安排紧凑、合理，能按不同部件、不同设备的实际需要分配时间；其缺点为：控制和设计复杂。

异步总线的应用场合主要为：异步系统总线，设备间工作速度差异大、传送时间不能预先确定或传送距离远的场合。

2．如何结合同步总线和异步总线的优点，使系统总线的设计控制简单又能合理安排时间？试举例说明。

可采用同步扩展方式。

例 1，在总线操作中，常规总线操作划分为 4 个时钟周期 T_1、T_2、T_3、T_4，如在 T_3 结束之前传送操作还没完成，可以在 T_3 和 T_4 之间插入一个或多个延长等待周期 T_W，但每个 T_W 的长度固定为时钟周期的长度，等传送操作完成后，才结束整个总线周期。这样总线的周期长度可以根据需要变化，实现了按需分配时间，时间安排合理，同时插入延迟时以固定的时钟周期长度为基准，使设计控制简单。

例 2，在同步方式中引入异步应答控制，在 PC 总线上有一种“三脉冲总线请求应答方式”，即在同一根总线上通过发出请求、响应、释放三个脉冲来实现总线权的转移，如图 3.5.1 所示。

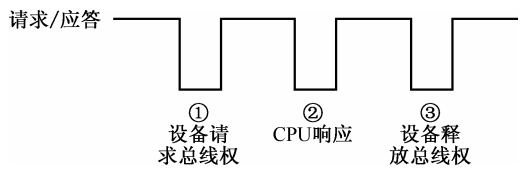


图 3.5.1　三脉冲请求应答方式

当某设备申请使用总线时，在请求/应答线上向 CPU 发出请求信号①，如 CPU 此时正在使用总线，则经过一到几个时钟周期，CPU 通过同一条信号线，向该设备发出响应信号②，从下一时钟周期开始 CPU 放弃总线权。设备使用完总线后，通过同一信号线向 CPU 发总线释放信号③，CPU 接管总线权。

在以上控制方式中，从信号①到信号②，再由信号②到信号③之间的时间长度是时钟周期的整数倍。

3．总线与设备之间的连接设备为接口，总线接口分为哪几种类型？

总线接口的分类可以从几个角度来划分：

（1）按数据传送格式划分，可分为串行接口和并行接口。

（2）按时序控制方式划分，可分为同步接口和异步接口。

（3）按数据传送控制方式划分，可分为直接程序传送方式接口、中断控制接口和 DMA 接口。

4．某机连接一输入设备，以中断方式向主机传送数据，试说明：

（1）该输入设备在何时提出中断？

当该输入设备的接口将输入数据转换为主机能识别的信息即数据已准备好，并已存放在接口的数据缓冲寄存器中时，接口向主机提出中断请求。

（2）画出该接口的寄存器级逻辑设计粗框图

该接口的寄存器级逻辑设计粗框图如图3.5.2所示。

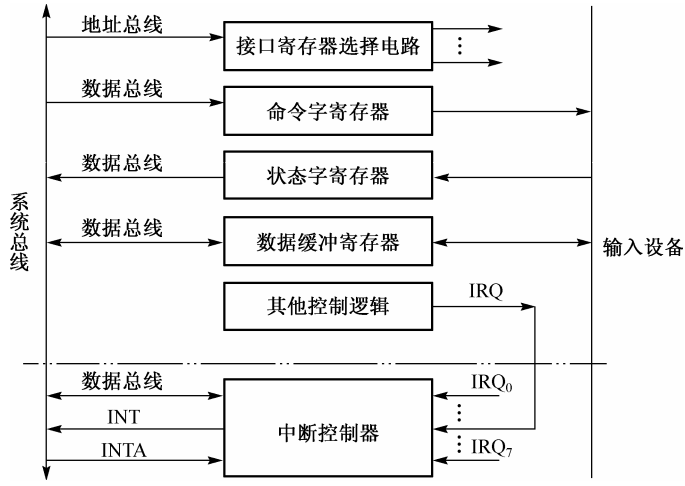


图 3.5.2　输入设备的中断接口图

（3）以向量中断工作方式描述中断从请求到处理的全过程

中断的全过程为：中断控制器汇集中断请求，经屏蔽、判优后，由中断控制器向 CPU 发出一公共中断请求 INT。

CPU 接到中断请求后，在一条指令执行结束后，向中断控制器发出中断响应信号 INTA，进入中断周期。

中断控制器接收到中断响应信号后通过数据总线送出该输入设备的中断类型码。

CPU 进入中断周期，关中断，保存断点，根据中断类型码计算向量地址，访问向量表获取中断服务程序的入口地址，转中断服务程序。

在中断处理中，首先保护现场，执行该输入设备的具体中断服务程序，发中断结束信号，恢复现场和断点，开中断，返回原程序执行。

（4）该设备的中断服务程序的功能是什么？

因为该设备为输入设备，所以该设备的中断服务程序的功能就是从接口的数据缓冲寄存器中读入数据至主机。

5．以打印机接口为例，试说明：

（1）输出设备在何时向主机提出中断请求？

第一次提出中断是在主机启动打印机，打印机完成启动准备好接收数据时向主机提出中断请求。以后在打印机打印完一次接收到的数据时（如通常为打印完一行数据），向主机提出中断请求。

（2）输出设备的中断服务程序的功能是什么？

输出设备，如打印机的中断服务程序的功能是将主机的数据传送至输出设备接口的数据缓冲寄存器中。

6．如某中断系统只允许单重中断，试说明中断处理的过程。

在单重中断中，CPU 进入中断周期时，关中断、保存断点，进入中断处理服务程序后其处理过程为：

（1）保护现场；

（2）执行具体的中断处理服务程序；

（3）发中断结束命令；

（4）恢复现场；

（5）开中断；

（6）返回。

7．如某中断系统允许多重中断工作方式，试说明中断处理的过程。

在多重中断工作方式中，CPU 进入中断周期后，关中断、保存断点，进入中断处理服务程序后其处理过程为：

（1）保存现场；

（2）送新屏蔽字，屏蔽与本中断源同级和优先级更低的中断源；

（3）开中断；

（4）执行具体的中断处理服务程序；

（5）发中断结束命令；

（6）关中断；

（7）恢复现场及原屏蔽字

（8）开中断；

（9）返回。

8．某机连接 4 台外部设备，如主机通过一个公共接口和中断类型码对它们进行初始设置和控制，这 4 台外设可向主机提出中断请求，试设计该中断接口，画出寄存器级粗框图。如主机要通过不同的中断源对设备进行控制，试提出一种中断源扩展的方案。

（1）如主机通过一个公共接口和中断类型码对设备进行控制，并接收外部设备的中断请求，则在接口中的命令字寄存器和状态字寄存器可由 4 台设备共用，将命令字寄存器和状态字寄存器划分为 4 段，分别对 4 台外部设备进行控制和反映设备状态。

当外设提出中断请求时，如 CPU 响应，CPU 根据公共的中断类型码转入中断服务程序。中断服务程序需先执行一段查询程序来确定具体是哪台设备提出的请求，再转该设备的中断服务程序。其寄存器级粗框图如图3.5.3所示。

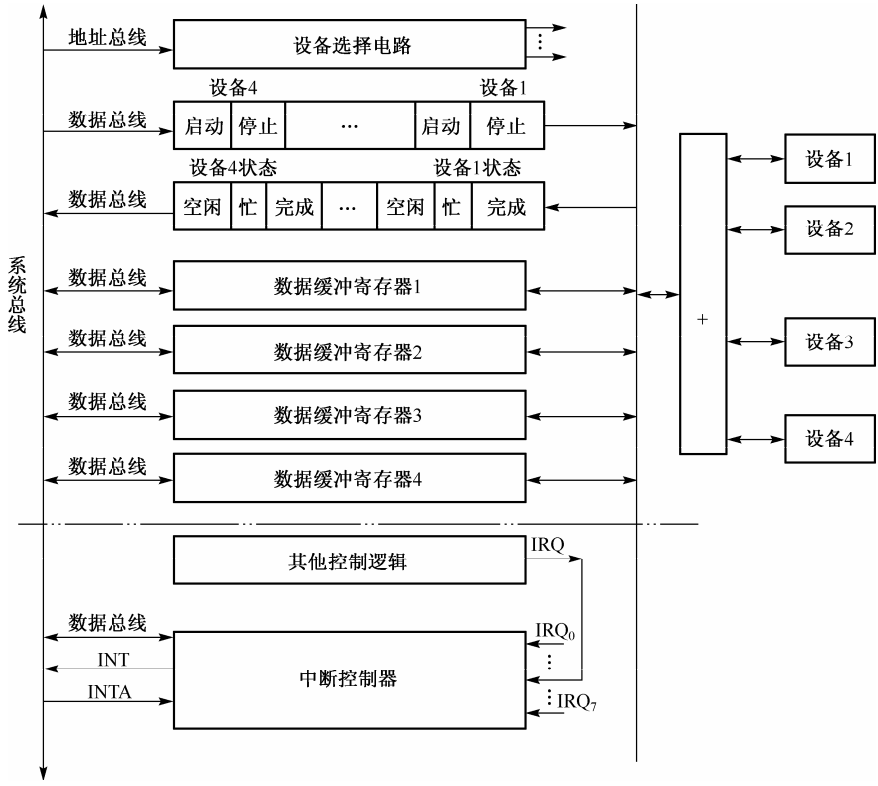


图 3.5.3　采用公共中断源的接口逻辑图

（2）如各设备需占用不同的中断源，当不同的设备提出中断请求时，不需执行查询程序，根据不同的中断类型码，CPU 就可以获得中断源，转相应的中断服务程序。但一般主机的中断控制器的中断源有限，所以需对中断源进行扩展，可以采用两个 8259 级连的方式来扩展中断源。两级中断控制器级连的方式如图3.5.4所示。

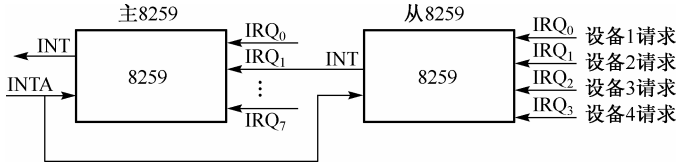


图 3.5.4　采用独立中断源的连接方式

扩展了中断源后，在系统初始化时，修改中断向量表，将扩展 8259 的中断源的中断向量填入向量表中。这样，当有设备提出中断请求后 CPU 响应时，就可直接根据中断向量转向的中断服务程序。

9．试描述 DMA 方式工作的全过程，并说明在 DMA 工作方式下哪些阶段需要 CPU 程序的干预？

DMA 方式是直接依靠硬件实现主存和 I/O 设备间的数据直传，传送期间不需要 CPU 的程序干预，DMA 的工作过程分三个阶段：

（1）DMA 初始化阶段：CPU 通过执行程序，向 DMA 控制器和接口提供如下初始化信息：

① 传送方向：即该次数据传送对主存而言是输入还是输出。

② 主存缓冲区首址：在主存中开辟的用于与 I/O 设备交换数据的缓冲区的首地址，写入 DMA 控制器的地址计数器中。

③ 交换量：本次数据传送的数据量，如字节数、字数或数据块数，用于控制传送的结束。

④ 外设寻址信息：外部设备的有关寻址信息。

（2）数据直接阶段：由 DMA 控制器控制发出地址信息，控制数据在主存和 I/O 设备之间的传送，并修改交换量计数。

（3）结束处理阶段：CPU 通过中断方式进行结束处理，主要对传送期间产生的错误进行处理。

由以上工作过程可知：在 DMA 的初始化阶段和结束处理阶段需要 CPU 执行程序干预。

10．以磁盘调用为例，说明在读磁盘和写磁盘时，磁盘适配器分别在什么时候向主机提出 DMA 请求。

在读磁盘时，当磁盘适配器的扇区缓冲器有一个扇区缓冲区装满时，表明适配器准备好 DMA 传送，这时磁盘适配器向主机发出 DMA 请求。

在写磁盘时，当扇区缓冲器有一个扇区缓冲区为空时，表明磁盘适配器已准备好接收数据，这时磁盘适配器向主机的 DMA 控制器发出 DMA 请求。

11．试比较中断方式和 DMA 控制方式的异同点。

中断方式和 DMA 控制方式的相同点是：都能随机产生请求，用于对随机事件的处理。

中断方式和 DMA 控制方式的不同点如下：

① 中断控制方式是通过执行中断服务程序处理主机与外设的信息交换，DMA 方式是直接依靠硬件实现主存与外设的数据直传。

② 中断控制方式可以处理复杂的事态，而 DMA 方式一般只用于简单的数据直传。

③ 由于中断控制方式是通过软件实现的数据传送，所以一般用于控制中低速的 I/O 操作。而 DMA 适用于高速的数据批量传送。

3.5.2 第 5 章习题解答与解题思路

1．简要解释下列名词术语。

【答】

系统总线：计算机系统内各功能部件之间，或各插件板之间互连的总线。

外总线：计算机系统之间，或计算机系统与其他系统之间互连的总线。

局部总线：直接与 CPU 连接的一段总线，称为局部总线。

通信总线：连接远距离信息传送的通信工具之间的传送线路，称为通信总线，常采用串行总线作为通信总线。

同步总线：数据的传送操作由统一的系统时钟同步定时的总线，称为同步总线。

异步总线：无固定时钟周期划分，总线周期时间由各部件操作的实际需要决定，采用异步应答方式控制传送操作的总线。

缓冲深度：接口中设置的数据缓冲寄存器的数量或缓冲区的容量，称为缓冲深度。

中断接口：如果主机与外围设备之间的信息传送采用程序中断方式控制，则接口需有相应的中断系统所需的逻辑，这样的接口称为中断接口。

DMA 接口：如果主机与高速外围设备之间的信息传送采用 DMA 方式控制，则接口中需有相应的 DMA 逻辑，这样的接口被称为 DMA 接口。

总线宽度：任何总线的线路在功能上可分为 3 组：数据线、地址线和控制线。所谓总线宽度，即各功能组中的信号线数。

主设备：申请并得到总线控制权的设备，称为主设备。

从设备：响应主设备请求的设备，称为从设备。

直接程序传送方式：CPU 直接利用 I/O 指令编程，实现数据的输入和输出的方式，称为直接程序传送方式。

中断：CPU 暂时中断现行程序的执行，转去执行处理某些随机事态的中断服务程序，处理完后自动恢复原程序的执行。

软中断：由执行软中断指令所引起的中断。

实时处理：指在事件出现的实际时间内及时地进行处理。

硬件中断：指由某个硬件中断请求信号引发的中断。

可屏蔽中断：CPU 可以通过送屏蔽字或开/关中断来禁止和开放中断，中断请求能否响应由 CPU 开/并决定。

非屏蔽中断：中断请求的响应不受屏蔽字影响，与 CPU 的开/关中断状态无关。

向量中断：将各个中断服务程序的入口地址（或包括状态字）组织成中断向量表；响应中断时，由硬件直接产生对应于中断源的向量地址；据此访问中断向量表，从中读取服务程序入口地址，由此转向服务程序。

非向量中断：以软件查询的方式提供中断服务程序入口地址的中断。

中断向量：中断服务程序入口地址和状态字在一起，称为中断向量。

向量地址：访问中断向量表的地址码，即读取中断向量所需的地址。

中断向量表：用来存放中断向量的一种表格，称为中断向量表。

中断隐指令操作：在中断周期中，直接依靠硬件实现关中断、保存断点、通过中断类型码计算中断程序的入口地址并转中断服务程序等操作，称为中断隐指令操作。

现场保护：执行中断服务程序时，可能使用某些寄存器，这就会破坏它们原先保存的内容，因此需要事先将它们的内容保存起来，称为现场保护。

DMA 方式：即直接存储器访问，DMA 方式是直接依靠硬件实现主存储器和外部设备间进行数据传送，传送时不需要 CPU 的干预。

DMA 初始化：在 DMA 传送操作开始前，为实现有关控制，CPU 需要事先向 DMA 控制器送出有关控制信息。在调用 I/O 设备时，通过程序所做的这些准备工作，称为 DMA 初始化。

2．比较并说明下述几种 I/O 控制方式的优缺点及其适用场合。

（1）直接程序传送方式（含程序查询方式）。

（2）程序中断方式。

（3）DMA 方式

【答】

（1）直接程序传送方式的优点：在直接程序传送方式中，CPU 只需实现 I/O 指令功能，不需要增加 CPU 硬件，因此硬件简单、控制简单。其缺点为：CPU 与外围设备完全不能并行地工作，CPU 利用率低，实时处理能力低。适用场合：CPU 速度不是很高，效率问题不是很重要的场合。

（2）程序中断方式的优点：由于中断是通过执行程序进行对事件的服务处理，处理程序可以根据需要进行扩展，所以程序中断方式的处理能力很强，可以处理复杂事态，随机性和实时性强。其缺点为：为了实现程序切换，决定 CPU 在中断周期中要执行隐指令操作：保存断点、读取服务程序入口地址，以及在转入服务程序后先应执行现场保护等操作；在返回原程序前，还需恢复现场、读取返回地址等。这一系列的操作要花费一定的时间，使中断方式难于适应高速数据传送。适用场合：适用于处理中、低速的 I/O 操作与随机请求，所处理的对象可以是复杂的随机事态。

（3）DMA 方式的优点为：可以响应随机请求，传送的实现是直接由硬件控制，CPU 不必为此执行指令，其程序不受影响，DMA 方式仅需占用系统总线，不需切换程序，因此不存在保存断点、保护现场、恢复现场、恢复断点等操作。其缺点为：仅仅依靠硬件，只能实现简单数据传送，难于识别与处理复杂事态。适用场合：适用于主存与高速 I/O 设备间的简单数据传送。

3．某机带四路数据采集器，以直接程序传送方式实现数据的输入。请为此

（1）设计接口，画出寄存器级粗框图。

（2）拟定相关的程序框图。

【答】

（1）设计分析：四路数据采集设备共用一个命令/状态寄存器，每一采集器的命令/状态字占用 2 位；当不需要接口工作时，通过编程设置使 D 与 B 均为 0，启动采集器工作时，CPU 通过命令使 D = 0，B=1，表示数据采集器开始工作；当采集器的数据准备好后设置 B = 0，D = 1 表示数据采集完成，CPU 可通过输入指令读入数据。

为了保证数据正确，四路数据采集器的数据缓冲寄存器分别设置。采集的数据分别放入各自的缓冲器中。其框图如图3.5.5所示。

（2）程序框图如图3.5.6 所示。

4、某机连接 4 台 I/O 设备，序号为 0～3，采用软件查询确定其中断优先级。请分别按下列两种要求拟定查询程序流程图。

（1）固定优先级。

（2）轮流优先，使机会均衡。

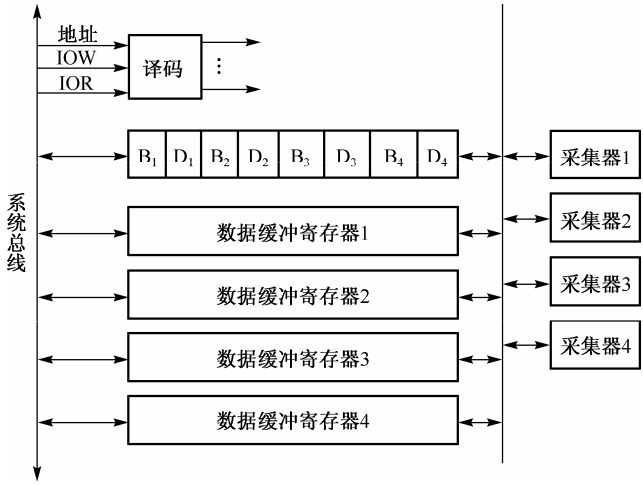


图 3.5.5　直接程序传送方式接口图

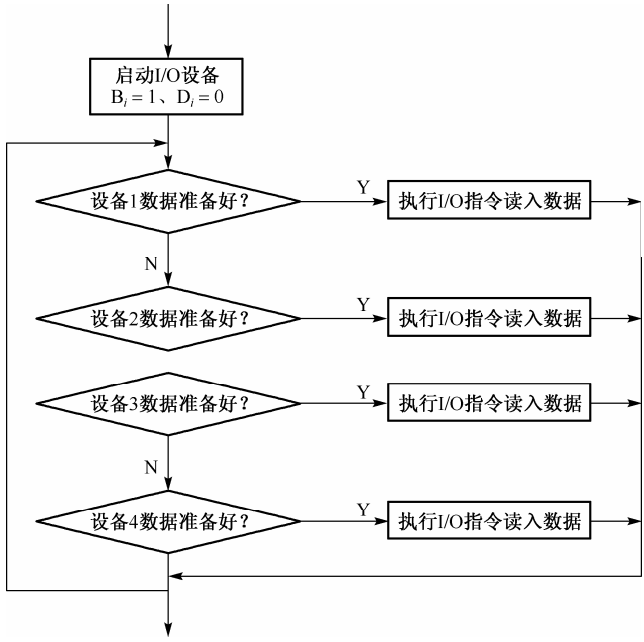


图 3.5.6　直接程序传送方式流程图

【答】

（1）在固定优先级方式中，CPU 响应中断请求后，先转入查询程序，按优先顺序依次询问各中断源是否提出中断请求，如果是，则转入相应的服务处理程序。假设在公共接口中设置一个中断请求寄存器，用来存放各中断源提出的请求信息，如对应位为 1 表示该中断源提出了请求。如设置为连接在最高位的中断源的优先级最高，依次优先级逐位降低。在进行软件查询时，先将中断请求寄存器的内容取回 CPU，然后依次判断，其程序流程如图 3.5.7 所示。

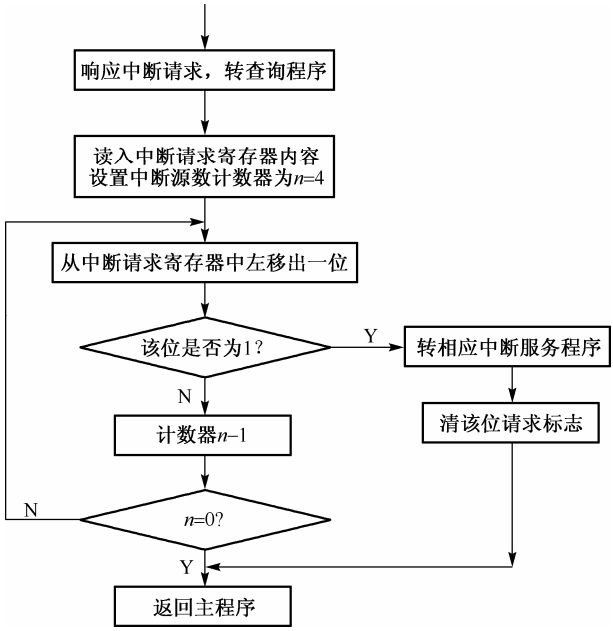


图 3.5.7　固定优先级流程图

（2）轮流优先

在轮流优先方式中，需设置中断请求寄存器，用于记忆当前有哪些中断源提出了中断请求。当某中断源提出中断请求后，中断请求寄存器的相应位设置为 1。另需设置记忆中断优先级状态的寄存器，描述各中断源的优先级，设优先权的等级为 3 时最低，为 0 时最高。如 $0^{\#}$~$3^{\#}$设备的初始优先级状态为 0、1、2、3，即：

	设备 $3^{\#}$	设备 $2^{\#}$	设备 $1^{\#}$	设备 $0^{\#}$
优先级状态	3	2	1	0

查询程序的流程如图 3.5.8 所示。

5．分别对图 3.5.9 所示两种结构，设计优先链排队逻辑，画出门级逻辑电路。

【答】

（1）针对多重查询方式其优先链排队逻辑图如 3.5.9 所示，在上半部分由门 1～门 6 组成一个串行的优先链，优先顺序从高到低为 $INTR_1$、$INTR_2$、$INTR_3$。若要扩充中断源，可根据其优先级别的高低串接于优先链的左端或右端。

图 3.5.9 的下半部分是一个编码电路，由总线向 CPU 发出被选中的设备码以供判定中断源。以下简单说明其工作过程，$INTR_i$ 为设备的中断请求信号，$INTS_i$ 为设备的中断排队选中信号。INTA 为中断许可信号。INTI 为中断排队输入，INTO 为中断排队输出信号。

当一个设备提出中断请求时，它将提供两个信号：$\overline{\text{INTR}}$ 在链中以低电平封锁优先级别比它低的请求；如果请求被选中是，它的 INTI 在链中以高电平打开自己的编码门，以提供设备码。

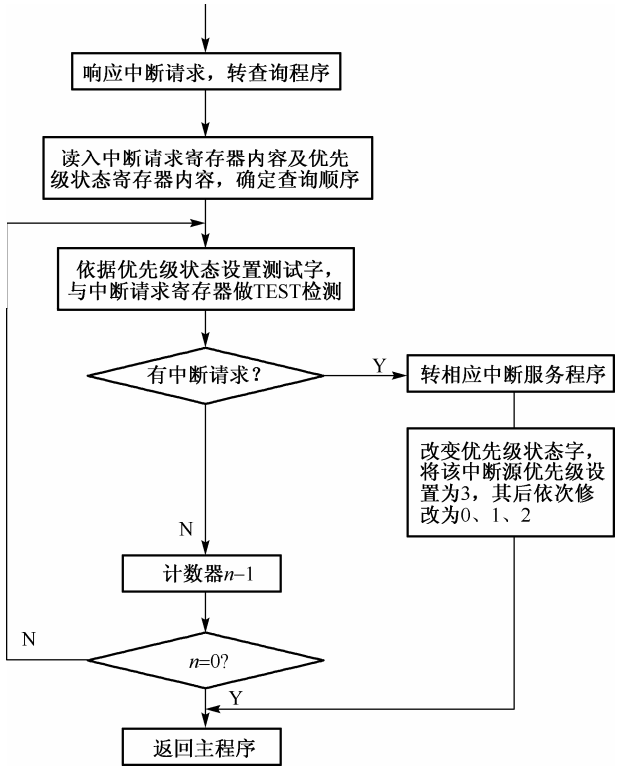


图 3.5.8　轮流优先级流程图

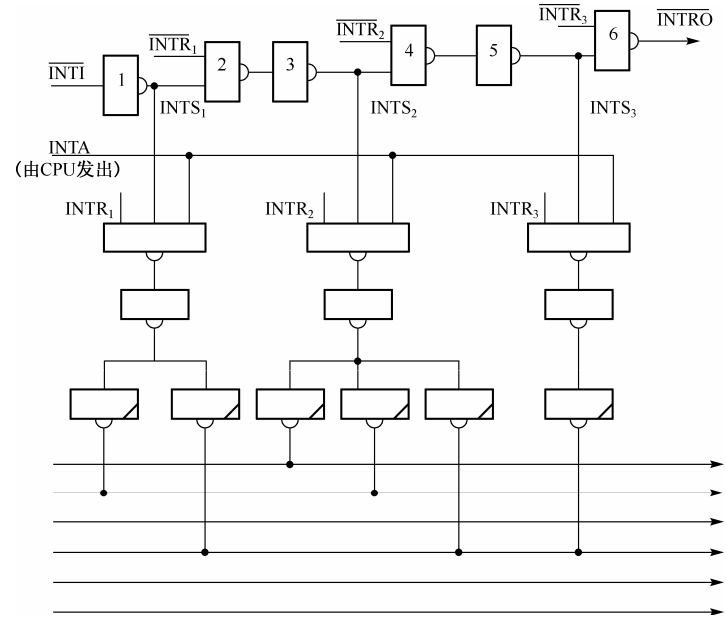


图 3.5.9　多重查询方式式判优逻辑图

可以分成三种情况讨论线路的状态。

① 若优先级高的 I/O 设备无请求时，则对优先级低的 I/O 设备请求开放。例如，没有比

本优先链中更高级别的请求时，则 $\overline{\text{INTI}} = 0$，门 1 的输出为“1”（高电平），即 $\text{INTS}_1 = 1$，允许设备 1 被选中。

② 若设备 1 此刻无中断请求，$\text{INTR}_1 = 1$，门 2 输出为“0”，则 $\text{INTS}_2 = 1$，就允许设备 2 被选中。若优先级高的 I/O 设备有请求，$\text{INTR}_1 = 0$，门 2 输出为“1”，则 $\text{INTS}_2 = 0$，$\text{INTS}_3 = 0$，即设备 2，设备 3 的请求都被封锁。

③ 若有两个以上的 I/O 设备同时提出请求，则只有优先级别高的 I/O 设备被选中。如设备 1 和设备 2 同时提出请求，当 CPU 接到请求信号并响应时，可通过程序发出一个中断询问命令查询是谁被选中，此时只有 $\text{INTS}_1 = 1$，而 $\text{INTS}_2 = 0$，所以是设备 1 被选中，取回该设备的设备码。

由上得出，经过链式排队，谁的优先级别高谁就向主机提供自己的设备码，主机就执行它的服务程序。

（2）若以菊花链式判优，则其连接逻辑如图3.5.10所示。

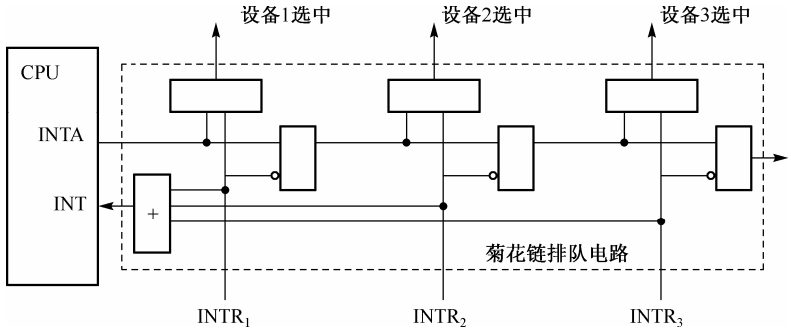


图 3.5.10　菊花链式判优逻辑图

6．程序中断方式与一般所指的转子有何不同？

【答】

中断具有随机性特点，子程序则没有，子程序的执行由程序员事先安排,而中断服务程序的执行则是由随机中断事件触发。子程序的执行受主程序或上层程序控制,而中断服务程序一般与被中断的现行程序无关。一般不存在同时调用多个子程序的情况,但可能发生多个外设同时向 CPU 发出中断服务请求的情况。

7．什么是向量中断方式和非向量中断方式？各有什么优缺点？

【答】

非向量中断：将所有中断源的中断服务程序入口地址组织在公共的查询程序中；CPU 响应时执行此查询程序，确定中断源对应的服务程序入口地址，再转入相应服务程序执行。

向量中断：将所有中断源的中断服务程序入口地址(中断向量)组织在中断向量表中；CPU 响应时由硬件产生向量地址，据此查表确定服务程序入口地址，再转入相应服务程序执行。

非向量中断通过软件来确定中断源，可以简化硬件逻辑，便于灵活修改中断源优先级，但非向量中断的响应速度比向量中断更慢。

8．现将模型机用于实时控制，控制 4 台电热炉，每台电热炉都有可能提出中断请求。试提出一种扩展中断输入的方案，并说明其响应、识别过程。

【答】

将 4 台电热炉共用一个中断源和中断类型码，其中断输入的扩展方式如图3.5.11 所示。

当有电热炉提出中断请求时，该请求通过 IRQ_1 送中断控制器，CPU 向中断控制发回响应信号，CPU 执行 IRQ_1 的中断服务程序，在 IRQ_1 的中断服务程序中，执行查询程序判断中断源并转具体的处理程序。

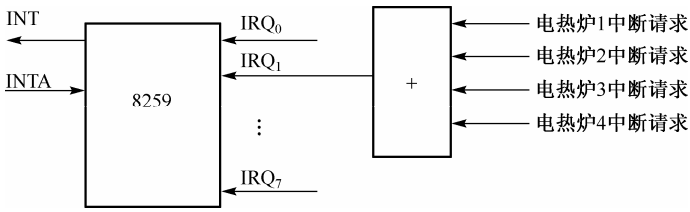


图 3.5.11　中断源扩展连接图

9．某机连接 4 台 I/O 设备，序号由 $0^{\#}$～$3^{\#}$，允许多重中断。

（1）优先顺序为 $0^{\#}$～$3^{\#}$，分别拟定响应各设备请求后应送出的屏蔽字。

（2）若采取轮流优先策略，则屏蔽字应如何变化？

【答】

（1）在该中断系统中，由于允许多重中断，在第一种情况下，当某个设备的请求得到响应后，应屏蔽比该中断源优先级低和同级的设备中断请求，而开放比该中断源优先级高的设备的中断请求信号。根据题意这 4 台设备的优先顺序分别为 $0^{\#}$～$3^{\#}$，当要屏蔽某中断源的中断请求时，需将该中断源的屏蔽位置为“1”，所以响应各设备请求后，应送出的屏蔽字分别为 0111、0011、0001、0000。

（2）若采取轮流优先的策略，则当某设备的中断请求得到响应后，将把比该设备优先级低一级的设备优先级置为最高，所以在轮流优先中响应各设备请求后，应送出的屏蔽字应分别为 1011、1101、1110、0111。

10．查阅有关资料，简要介绍 ISA 总线的信号组成。

【答】

ISA（Industry Standard Architecture）即工业标准体系结构总线。ISA 总线是 8 位/16 位兼容的总线，其信号线由 98 根线组成，可分为前 62 线段和后 36 线段，其信号组成为：16 位数据线、24 位地址线、控制线、状态线和其他信号线（如时钟 OSC/CLK 信号线、±12V、±5 V 电源线和地线等）。

控制线包括：

ALE：地址允许信号，输出线，高电平有效，表明正处于 DMA 控制周期中，此信号可用来在 DMA 期间禁止 I/O 端口的地址译码。

ALE：允许地址锁存，输出线，该信号由 8288 总线控制器提供，作为 CPU 地址的有效标志，其下降沿用来锁存 A_0～A_{19}。

$\overline{\text{IOR}}$：I/O 读命令，输出线，用来把选中的 I/O 设备的数据读到数据总线上。在 CPU 启动的 I/O 周期，通过地址线选择 I/O；在 DMA 周期，I/O 设备由 $\overline{\text{DACK}}$ 选择。

$\overline{\text{IOW}}$：I/O 写命令，输出线，用来把数据总线上的数据写入被选中的 I/O 端口。I/O 端口的选择方法与 $\overline{\text{IOR}}$ 所述相同。该信号由 CPU 或 DMA 控制器产生，经总线控制器 8288 送至总线。

$\overline{\text{SMEMR}}$：存储器读命令，输出线，用来把选中的存储单元中的数据读到数据总线上，该信号的产生与 $\overline{\text{IOR}}$ 相似。用于对 20 位地址寻址的 1 MB 内存的读。

$\overline{\text{SMEMW}}$：存储器写命令，输出线，用来把数据总线上的数据写入被选中的存储单元。用于对 20 位地址寻址的 1 MB 内存的写。

$\overline{\text{MEMR}}$ 和 $\overline{\text{MEMW}}$：存储器读和写命令，这两个选通线对全部存储空间有效。

$\overline{\text{MEM CS 16}}$（存储器 16 位片选信号）和 $\overline{\text{I/O CS 16}}$（I/O 16 位片选信号）：这两个信号分别指明当前数据传送是 16 位存储周期和 16 位 I/O 周期。

$\overline{\text{MASTER}}$：输入信号，由希望占用总线的有主控能力的外设适配器驱动。

T/C：DMA 终止计数，输出线，该信号是一个高电平脉冲，表明 DMA 传送的数据已达到其程序预置的字节数，用来结束一次 DMA 数据块传送。

IRQ_2～IRQ_7、IRQ_{10}～IRQ_{15}：中断请求，输入线，用来把外部 I/O 设备的中断请求信号，经过系统板上的 8259A 中断控制器送给 CPU。

DRQ_0～DRQ_7：DMA 请求，输入线，用来把 I/O 设备发出的 DMA 请求（高电平）传至系统板上的 DMA 控制器，产生一个 DMA 周期。

$\overline{\text{DACK}}_0$～$\overline{\text{DACK}}_7$：DMA 响应，输出线，表明对应的 DMA 请求已被接受。DMA 控制器将占用总线并进入 DMA 周期。

SBHE：总线高字节允许信号，表示数据总线传送的是高位字节数据。

REFRESH：刷新控制。

RESET DRV：系统复位信号，输出线，为接口提供电源接通复位信号，使各部件置于初始状态，此信号在系统电源接通时为高电平，当所有电平都达到规定后变低。

状态线包括：

$\overline{\text{I/O CHCK}}$：通道检查，输入线，低电平有效用来表明接口插件或系统板存储器出错，它将产生一次不可屏蔽中断。

$\overline{\text{I/O CHRDY}}$：I/O 通道就绪，输入线，高电平表示“就绪”。该信号线可供低速 I/O 或存储器请求延长 CPU 的总线周期。

$\overline{\text{0WS}}$：零等待状态信号线，该线被拉成低电平时，通知微处理器当前总线周期能完成，无需插入等待周期。

11．查阅有关资料，简要介绍Multibus总线的信号组成。

【答】

Multibus 总线是 Intel 公司推出的一种总线标准，经 IEEE 标准化，称为 IEEE-796 标准总线。Multibus 既支持 8 位数据总线，也支持 16 位数据总线，如果是 32 位数据总线，则需使用 Multibus II。Multibus 的特点是：容易构成多处理机系统，任一时刻，总线上只能有两个设备

相互通信，一个作为总线主设备，另一个作为总线从设备。主/从关系是动态的，通过总线裁决器来确定新的主/从关系；Multibus 总线支持 8 级中断请求信号，信号线全部采用负逻辑。

Multibus 的信号组分为两组，分别位于插件板上的两个插头 P1 和 P2。P1 是主插头，提供主要的总线信号，供 86 条引脚；P2 是可选插头，供 60 条引脚，主要用来做掉电检测和掉电处理。

主要的信号组成包括如下 5 组。

（1）地址线和有关控制线

① P1 上有 20 根地址线，P2 上还有 4 根地址线，用于传送要访问的存储单元或 I/O 端口地址：

如为 8 位总线主设备，则用 16 条地址线寻址存储器，用 8 条地址线寻址 I/O 端口。

如为 16 位总线主设备，则用 20 条地址线寻址存储器，用 12 条地址线寻址 I/O 端口。

②禁止线 $\overline{INT_1}$ 和 $\overline{INT_2}$。

$\overline{INT_1}$：RAM 禁止信号，阻止 RAM 对地址总线上的地址做出响应。

$\overline{INT_2}$：ROM 禁止信号，阻止 ROM 对地址总线上的地址做出响应。

提供以上两个禁止信号的作用是，使公共地址空间的 RAM、ROM 以及辅助 RAM、ROM 的地址相互重叠。

（2）数据线和有关应答线

数据线：DAT_{15}～DAT_0（16 位系统中）

DAT_7～DAT_0（8 位系统中）

应答线包括：

$\overline{XACK}$：用于应答式的数据交换，是从设备对主设备的应答信号，表明规定的操作已经完成。它被送到产生 READY 信号的逻辑电路。

$\overline{BHEN}$：高位字节允许信号，有效时，表示高 8 位数据线 DAT_{15}～DAT_8 上的数据有效。

$\overline{MRDC}$：存储器读信号。

$\overline{MWTC}$：存储器写信号。

$\overline{IORC}$：I/O 端口读信号。

$\overline{IOWC}$：I/O 端口写信号。

（3）中断控制信号线

Multibus 提供 8 条中断请求信号线 $\overline{INT_7}$ ～ $\overline{INT_0}$ 和 1 条中断相应信号线 $\overline{INTA}$。

（4）总线仲裁控制信号

$\overline{BCLK}$：总线时钟信号。

$\overline{BPRN}$：总线优先级输入信号。

$\overline{BPRO}$：总线优先级输出信号。

$\overline{BUSY}$：总线忙信号。

$\overline{BREQ}$：总线请求信号。

$\overline{CBRQ}$：公共的总线请求信号。

（5）其他信号，如电源、地、复位等。

12．查阅有关资料，比较ISA总线与Multibus总线之间的异同。

【答】

ISA 总线和 Multibus 总线之间的相同之处为：

① 数据总线宽度：既支持 8 位总线又支持 16 位总线。

② 总线工作频率：8 MHz。

③ 数据传输率：16 MBps。

④ 同步方式：半同步。

⑤ 都不具有多路复用的功能。

其不同之外为：

① ISA 总线的地址线宽度为 24 位，可寻址空间可达 16 MB，Multibus 总线的地址线宽度为 20 位，可寻址空间 1 MB。

② 应用场合：ISA 总线主要用于微型计算机系统的系统总线，而 Multibus 主要用于 Intel 公司的 SBC 单板机，是工业控制领域常用的系统总线，也常用于板与板之间连接的总线。

13．某 I/O 设备的工作状态可抽象为空闲、忙、完成，CPU 发来清除命令使其进入空闲状态，启动命令使其进入“忙”状态，设备完成一次操作使其进入完成状态。若进入完成状态，且 CPU 没有对其屏蔽，则提出中断申请。试为此设计中断接口，画出逻辑图。

【答】

其中断接口如图3.5.12所示。

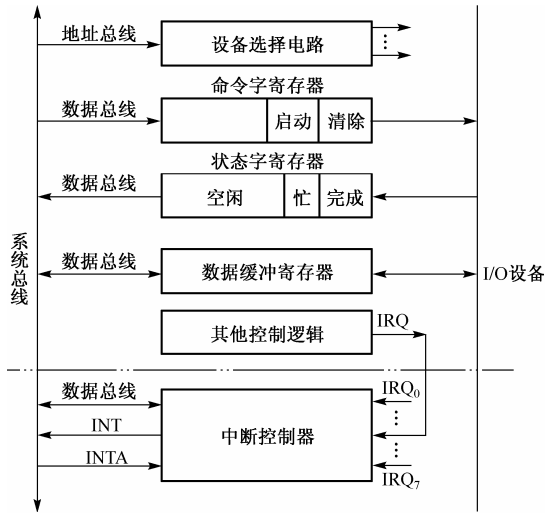


图 3.5.12　中断接口图

14．某机用于数据采集，采集点 64 点，共占一个中断源。CPU 在响应中断后能编程访问各采集点，输入数据。试为此设计所需的中断接口，并说明 CPU 选择采集点的方法。

【答】

在该系统中需要实现对 64 个采集点的数据采集，由于该 64 个点共占一个中断源，所以需采用 CPU 轮询的方式进行数据采集，其中断接口中命令字、状态字寄存器和数据缓冲寄存器的组成为：

64 个点共用一个命令字寄存器，分配不同的位来描述对各采集点发送的命令；

64 个点共用一个状态字寄存器；

64 个点分别有一个数据缓冲寄存器，分配不同的地址。

CPU 通过地址总线传送不同的地址至接口，根据地址的不同确定 CPU 选择的采集点。数据采集工作的基本过程为：CPU 向命令寄存器相应位置“启动”命令，并发送需采集点的地址，接口将对应采集点的状态位置为“忙”，当采集点将采集的数据放入其分配的数据缓冲寄存器后，接口将该采集点的状态置为“完成”，并向 CPU 提出中断请求，CPU 执行中断服务程序，由采集点的地址选择相应的数据缓冲寄存器读入数据。其中断接口组成的粗框图如图3.5.13所示。

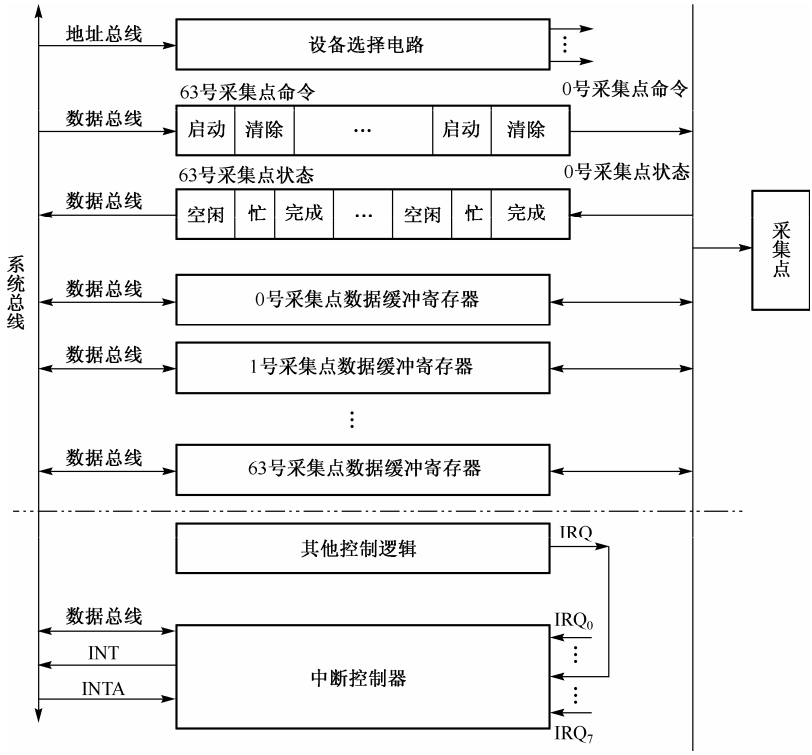


图 3.5.13　多数据采集点中断接口图

15．某机用于控制 8 层楼电梯系统，请根据你对电梯运行方式的了解，设计中断接口，画出寄存器级逻辑粗框图，并拟定其命令字与状态字格式。

【答】

用于电梯控制的命令字状态字格式如图3.5.14所示。

命令字寄存器中存放 CPU 控制电梯动作所发出的命令。当有人按下电梯按钮时，状态字寄存器中存放当前是哪一层有使用电梯的请求，及该请信号来自电梯内还是电梯外，分别用两个状态字寄存器存放梯内和梯外各层的请求。分别用两个数据缓冲寄存器存放电梯运行当前到达的位置和运行的速度。CPU 读入数据缓冲寄存器中的数据，与状态字寄存器中的请求状态做比较，向命令字寄存器中送控制命令，以确定当前是应该启动电梯，或是停止；是提

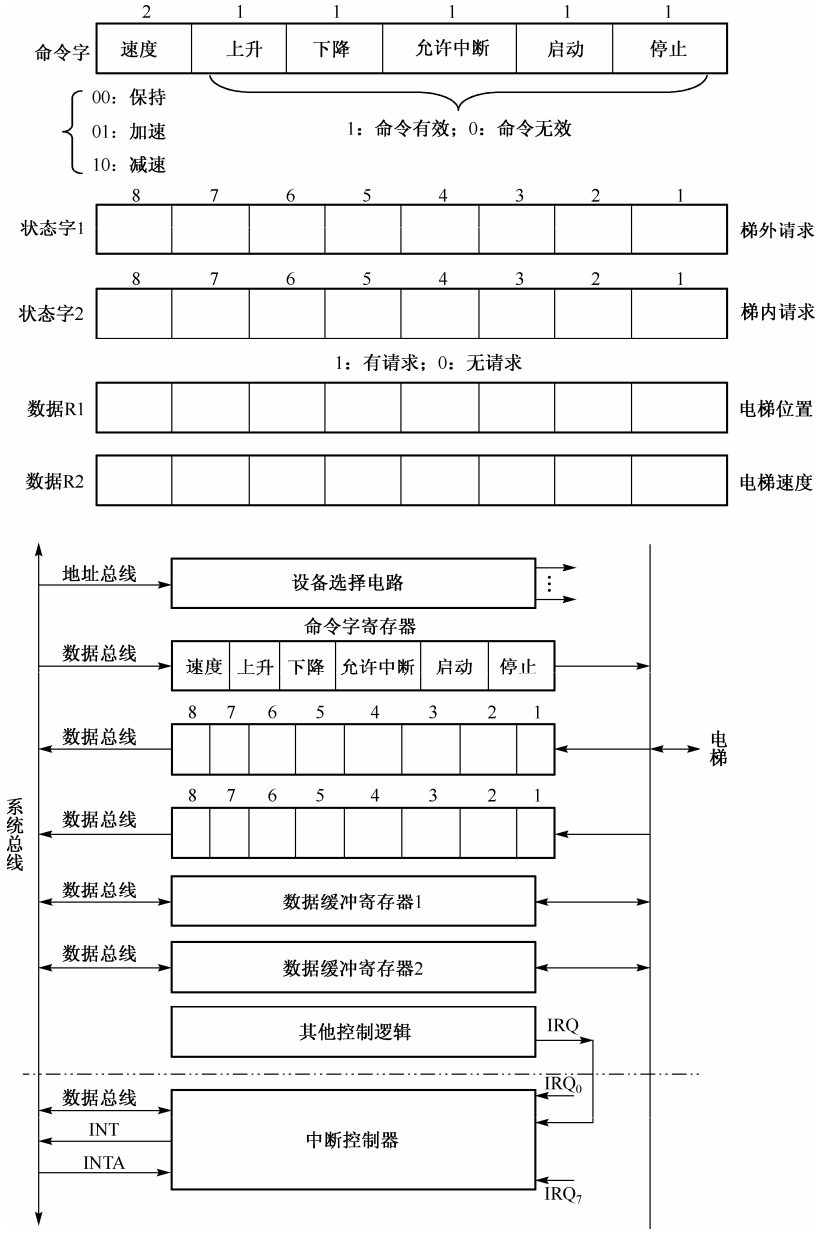


图 3.5.14　电梯控制中断接口图

速（从一层启动时）、减速（要到达有请求的楼层时）还是保持当前速度运行，是向上运动还是向下运动等。

16．某机连接 8 台炉温控制器，它们各控制一台电热炉。主机通过公共接口选择炉温控制器，对它进行初始化设置、启停控制，从控制器输入数据。炉温控制器在炉温异常或需与主机通信时，可向主机提出中断请求。试为此设计中断接口，画出寄存器级粗框图。

【答】

主机通过公共接口选择炉温控制器，则在接口中各炉温控制器共用控制寄存器和状态字寄存器，将控制寄存器/状态字寄存器划分为 4 段，分别用于对 4 个炉温控制器的设置和反映 4 台控制器及电热炉的状态。

各控制器各占用一个数据缓冲寄存器，用于存放从控制器输入的数据和主机传出的控制数据。该中断接口的寄存器级粗框图如图3.5.15所示。

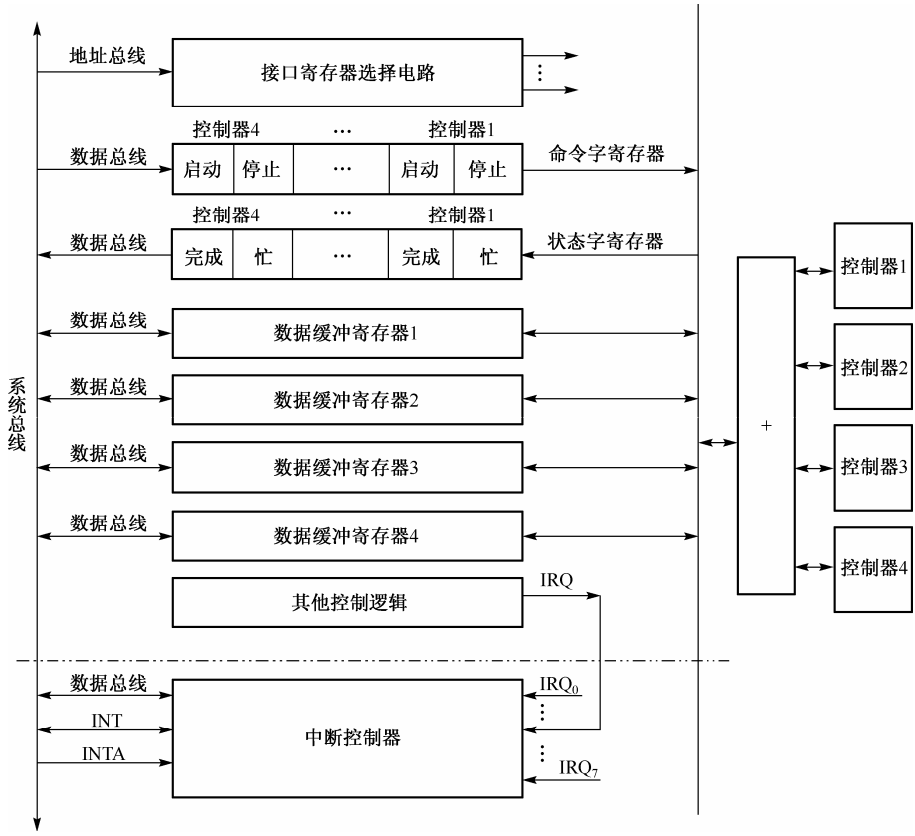


图 3.5.15　炉温控制器中断接口图

17．某机连接 3 台 I/O 设备，优先顺序为 CRT 显示器，打印机、键盘。显示器在扫描回程开始时提出中断，主机输出显示信息。打印机在打印完一行时提出中断，主机输出打印信息。键盘在按键时产生中断，向主机输入键码。请为此设计中断系统。

（1）说明硬件、软件组织的方案。

（2）画出有关的接口逻辑框图。

（3）拟出有关的程序流程。

【答】

（1）

硬件组织方案：

① 分别为 CRT 显示器、打印机和键盘设计各自的接口，通过接口将显示器、打印机和键盘连接至系统总线与主机相连。

② CRT 显示器、打印机和键盘接口上的中断请求信号通过总线送至系统板上中断控制器的中断请求源上。为每一接口分配一中断源，假设显示器、打印机、键盘分别占用中断源 IRQ_2、IRQ_3、IRQ_4。

软件组织方案：

① 系统初始化软件：初始化中断控制器：将设定好的优先级顺序写入 8259A 的优先级仲裁器。初始化向量表：保护源向量表中 IRQ_2、IRQ_3、IRQ_4 中断源的向量。分别将 CRT 显示器、打印机、键盘的中断服务程序入口地址写入 IRQ_2、IRQ_3、IRQ_4 中断源对应向量地址单元中。

② 分别为 CRT 显示器、打印机、键盘编写中断服务程序。

（2）各设备的接口

① CRT 显示器接口如图3.5.16所示。

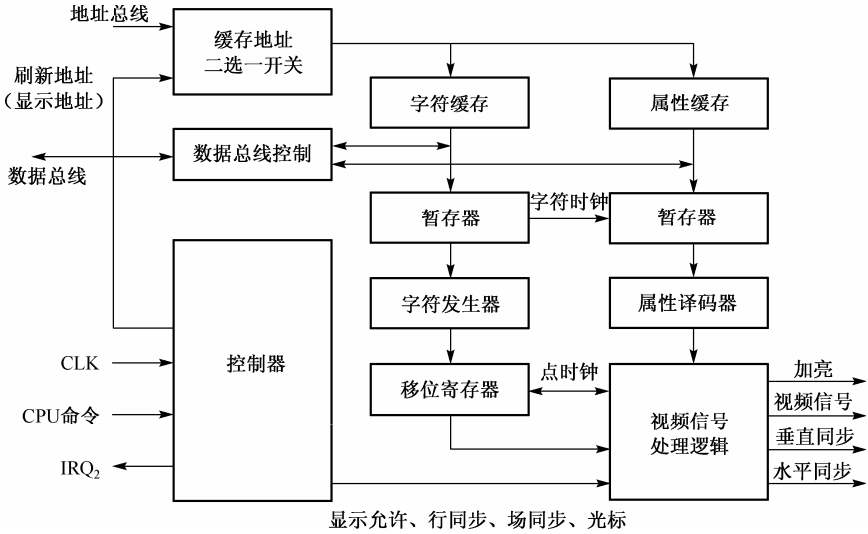


图 3.5.16　CRT 显示器接口图

② 打印机接口如图3.5.17所示。

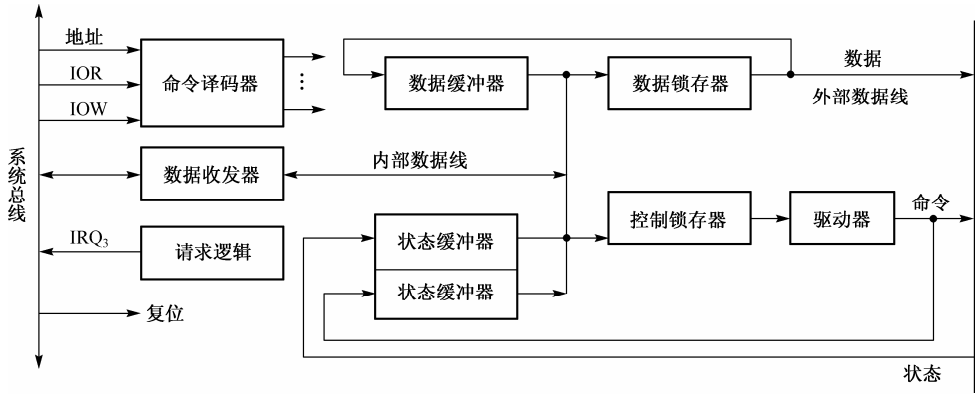


图 3.5.17　打印机接口图

③ 键盘接口如图 3.5.18 所示。

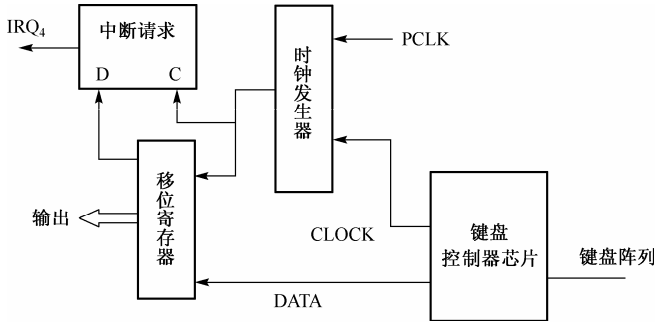


图 3.5.18　键盘接口图

（3）程序流程如图 3.5.19 所示。

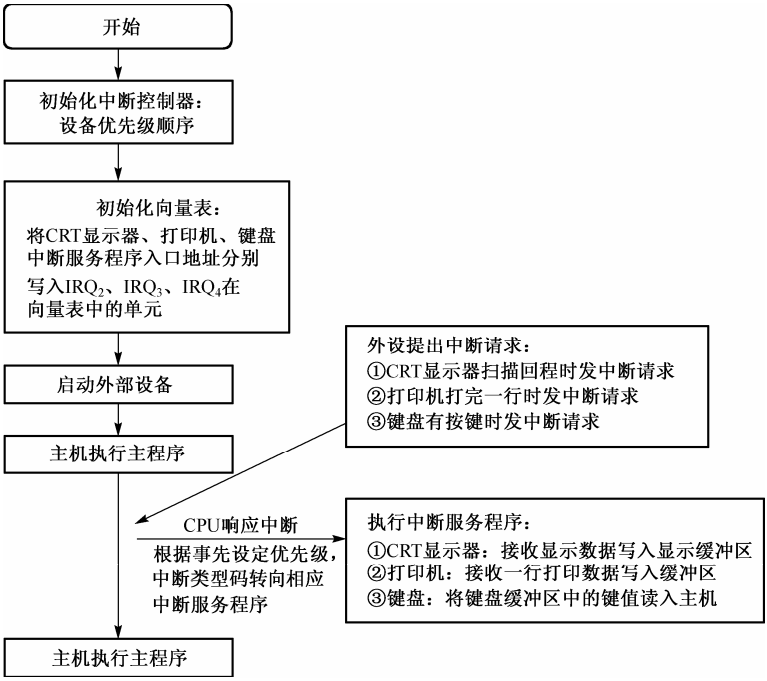


图 3.5.19　程序流程图

18. 某输入设备按串行方式工作，以向量中断方式向主机传送数据，CPU 发向设备的命令包括清除、屏蔽、启动、停止，设备状态可抽象为空闲、忙、完成、故障，试为该设备设计中断接口。

（1）设计中断接口，画出寄存器级粗框图。

（2）说明图中各组成部分的功能。

（3）拟定接口命令字格式和状态字格式。

（4）以设备向主机输入数据为例，描述向量中断过程，说明：① 主机如何启动设备？② 设备在什么情况下申请中断？③ 接口如何传送中断请求？④ CPU 响应后如何转相应服务程序？

【答】

（1）寄存器级粗框图如图 3.5.20 所示。

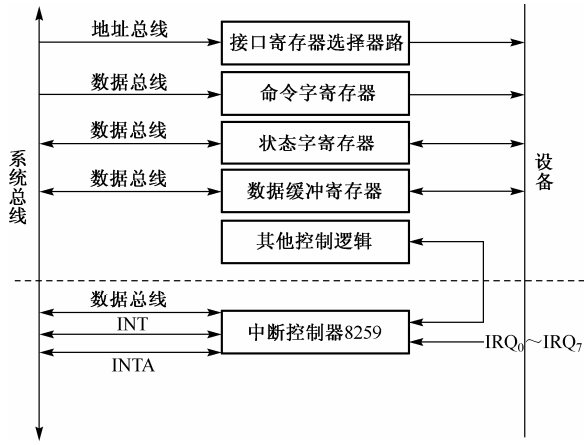


图 3.5.20　寄存器级粗框图

（2）图中各部分的功能如下。

接口寄存器选择电路：接收地址码，译码后产生选择信号，选择寄存器。

命令字寄存器：保存 CPU 发送给外设的控制命令。

状态字寄存器：记录、反映设备与接口的运行状态。

数据缓冲寄存器：对输入和输出的数据进行缓冲。

其他控制逻辑：产生请求信号、控制时序，进行串-并转换等。

中断控制器：汇集各接口的中断请求信号，经过屏蔽控制优先排队，形成送往 CPU 的中断请求信号 INT；接到 CPU 批准信号 INTA 后，通过数据总线送出向量地址（或中断类型码）。

（3）

命令字：

清除	屏蔽	启动	停止

状态字：

空闲	忙	完成	故障

（4）

① 通过发送启动命令来启动外设。

② 当设备准备好，或完成一次操作时向中断控制器发出中断请求。

③ 设备中断请求信号送中断控制器 8259A，经屏蔽控制和优先排队，向 CPU 发出公共请求 INT。

④ CPU 响应以后，向 8259A 发回批准信号 INTA，并通过数据总线从 8259A 取走对应的中断号转换为向量地址并访问中断向量表，读取中断向量，并据此作为入口地址转向中断服务程序执行。

19．在 DMA 的初始化阶段中，主要应完成哪些任务？

【答】

① 向接口送出 I/O 设备的寻址信息。

② 向 DMA 控制器送出控制字，如传送方向。

③ 向 DMA 控制器送出主存缓冲区首址。

④ 向 DMA 控制器送出传送的数据量。

20．DMA 控制器与接口的连接模式有哪几种？各有什么特点？

【答】

① 单通道 DMA 控制器：只能由一个外设独占使用，接口集成在控制器内部，数据传送要经过控制器。

② 选择型 DMA 控制器：可以被多个外设共享使用，由控制器选择其中一个设备进行工作，接口集成在控制器内部，数据传送经过控制器。

③ 多路型 DMA 控制器：接口独立于控制器，各设备有自己独立的接口，所有的接口均被控制器统一控制，数据传送不经过控制器。

21．某机以星形模式连接 4 台设备控制器，系统总线采用 PC 总线标准，与设备间分别采用串行通信接口。请为此设计一块四串口卡，画出寄存器级粗框图。

【答】

主机以星形模式连接 4 台设备的控制器，串行通信卡上与 PC 总线连接的一端采用一个统一的数据总线缓冲模块，4 个串口通过数据总线缓冲模块与总线之间收发数据。因为该串口卡要分别与 4 台设备的控制器相连，因此，串口卡上应有 4 套独立的数据收发控制和数据寄存器。其寄存器级粗框图如图3.5.21所示。

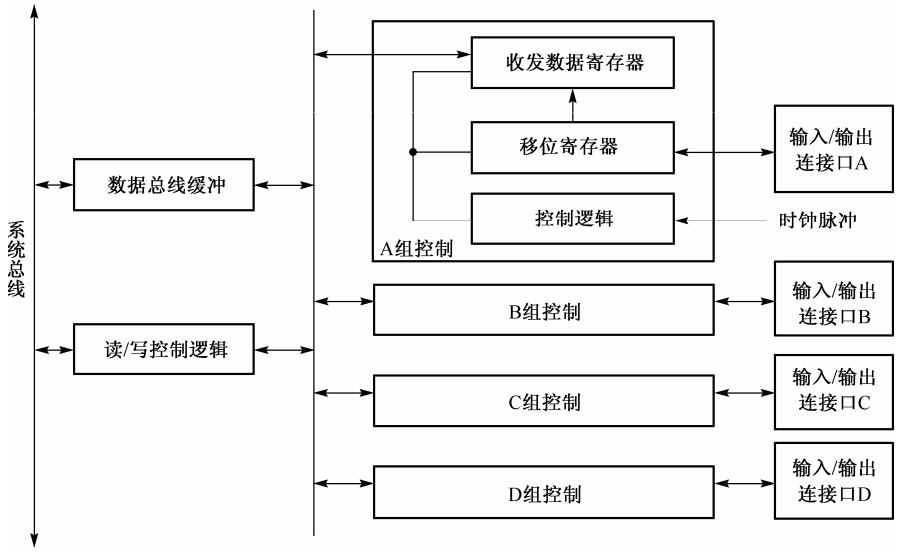


图 3.5.21　四串口卡逻辑图

说明：① 读/写控制逻辑用于接收 CPU 发来的端口地址与读/写命令，向各端口发出操作命令。

② 数据总线缓冲：一侧与数据总线连接，另一侧与接口内总的数据线连接，从系统总

线上获得数据，分送接口内各部件；或接收接口内总件，送往数据总线。

22. 某机通过一个通信卡连接一组串行通信总线，再通过串行总线连接多个设备控制器。通信卡的一侧面向 PC 总线，另一侧面向串行通信总线，通信卡为含有微处理器、局部存储器的智能型接口。设备控制器也是含有微处理器、局部存储器的智能型控制器。试为此设计通信卡，画出寄存器级组成的粗框图。

【答】

由该通信卡的功能可知，其逻辑上应由三部分组成。一部分是面向 PC 总线的一侧，应包含与总线交换信息的常规部件如，地址译码、命令字/状态字、输入/输出通路等；另一部分是面向串行通信总线的一侧，因要与串行通信总线相连，因有总线驱动、数据锁存等部件；中间部分应是含有微处理器、局部存储器和用于特定控制（如串行移位控制、中断产生、时序等）的相关控制逻辑部分，其寄存器级粗框图如图3.5.22 所示。

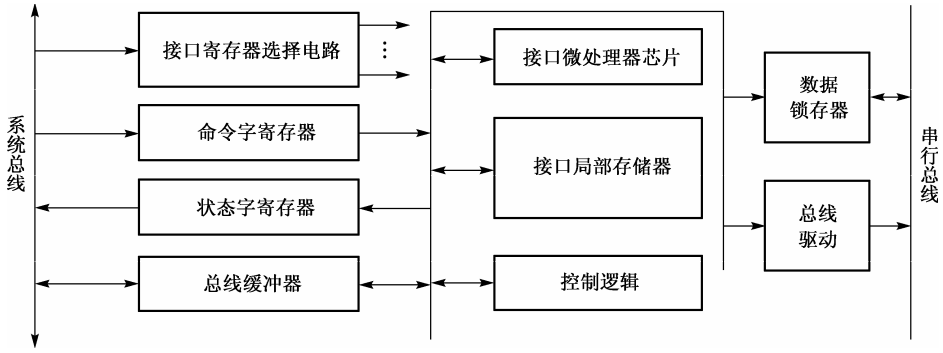


图 3.5.22　通信卡逻辑图

3.6　第 6 章习题解答及解题思路

3.6.1　第 6 章重点和难点解析

1．从计算机整体功能来看，I/O 设备具备哪些功能？

I/O 设备具备的功能可归纳为 4 方面：

① 在处理机与外界联系之间完成信息形式的转换。

② 实现人机交互。

③ 存储备用的系统软件及各种信息。

④ 通过 I/O 设备促进计算机应用领域的扩展。

2．显示器和显示适配器的显示规格主要指哪些？

显示规格主要有显示方式、分辨率和色彩三方面。

显示方式：是字符方式还是图形方式，或既可采用字符方式也可采用图形方式。

分辨率：字符方式时，通常用一帧画面最多可显示的字符列数和行数表示，如 80 列×25 行。

图形方式是用一帧画面最多可显示的像点数来表示，如 800 点×600 线。

色彩：是单色还是彩色。目前已广泛采用彩色显示，故能显示的颜色数量也是显示规格的内容之一。

3．在 CRT 字符显示器中，显示缓存和字符发生器各采用什么存储器来构成？各采用何种存取方式？决定它们存储容量大小的主要因素各有哪些？

显示缓存由 RAM 芯片构成，采用随机存取方式；显示缓存的容量主要取决于字符显示器的分辨率、要求显示的颜色数和显示属性的数量。字符发生器由 ROM 构成，采用只读方式工作；字符发生器容量主要取决于字符显示器可显示的字符种数和字符点阵规格。

4．若 CRT 字符显示器采用逐行扫描方式，帧频为 70 Hz；全屏能显示 80 列×25 行字符，字符点阵规格为 7（横向）×8（纵向），字符间隔 1 个像素点，字符行间隔为 4 条扫描线；若屏幕上、下边缘消隐区和垂直回扫扫描时间折合为 5 行字符，屏幕左、右边缘消隐区和水平回扫折合为 14 个点阵字符宽度。问：

（1）各同步计数器的位数和分频关系各为多少？

点计数器：每计 8 个点（包括点阵字符宽度 7 点和字符间隔 1 点），计数器状态回零，故点计数器应设为 3 位，分频关系为 8∶1。

字符计数器：每计 94 个字符（包含每条扫描线的 80 个字符和屏幕左、右边缘及水平扫描消隐段折合的 14 个字符），计数器状态回零，故字符计数器设为 7 位，分频关系为 94∶1。

线计数器：每计 12 条扫描线（包含一字符行的 8 条扫描线和字符行间间隔的 4 条扫描线），计数器状态回到零，故线计数器应设置为 4 位，分频关系为 12∶1。

行计数器：每计 30 行字符（包含一帧的 25 行字符和屏幕上下边缘及垂直回扫消隐区折合的 5 行字符），行计数器状态回到零，计数器位数应设置为 5 位，它的分频关系为 30∶1。

（2）各同步计数器的计数频率各为多少？

根据题目给定的条件，若字符显示器的帧频为 70Hz ，则各计数器的计数频率如下。

行计数器的频率 $F_{行}=30\times70=2.1(\text{kHz})$

线计数器的频率 $F_{线}=12\times F_{行}=12\times2.1=25.2(\text{kHz})$

字符计数器的频率 $F_{字}=94\times F_{线}=2368.8(\text{kHz})$

点计数器的频率 $F_{点}=8\times F_{字}=18950.4(\text{kHz})=18.9504(\text{MHz})$

5. 某 CRT 显示器分辨率为 660×270，将它作为字符显示器以显示 94 种 5×7 点阵字符，若字符之间的间隔为 1 个像素点，字符行之间的间隔是 3 条扫描线。问：

（1）该显示器全屏（即一帧）能显示多少行字符？

从分辨率可知，显示器一帧有 270 条扫描线，每个字符纵向为 7 个像素点，故一行字符要占用 7 条扫描线，字符行间间隔占 3 条扫描线，故全屏能显示的字符行数为：

$$\frac{270}{7+3}=27\ （行/帧）$$

（2）该显示器一行能显示多少个字符？

从分辨率可知，一条光栅有 660 个像点，每个点阵字符横向占有 5 个像点，字符间隔为一个像素点，故一行最多可显示的字符个数为：

$$\frac{660}{5+1}=110 \text{（字符/行）}$$

（3）一帧能显示多少个字符？

一帧能显示 110×27=2970 个字符。

（4）基本显示缓存（即不考虑显示属性及颜色）放什么？它的最小容量为多少？

基本显示缓存放一帧字符的 ASCII 码，一字节单元放一个字符的 ASCII 码，故基本显示缓存应能存放 2970 个字符的编码，所以它的最小容量应为 2970 字节。

（5）字符发生器放什么？字符发生器最小容量为多少？

字符发生器用于存放字符显示器所能显示的字符的点阵代码。理论上，字符发生器最小容量为 5×7×94=3290（位）。

通常，用于显示的字符点阵是按行（线）存放。本题中，1 个字符需 7 行（线）点，放 1 个字符的点阵代码需 7 字节单元（注：每线 5 个点，占 1 字节的 5 位，另 3 位空闲），于是字符发生器最小容量为 7×94=658（字节）。

（6）假设屏幕字符的行号、列号以及基本显示缓存的地址均从 0 开始计，现在屏幕上第 5 行第 6 列的字符 A 应放基本显示缓存的哪一个字节单元。

屏幕上的字符 A 应存放在基本显示缓存的$5\times110+6=556$单元。

（7）显示器的同步计数器应设几级?它们的分频关系如何？

同步计数器设置 4 级，按计数顺序依次为点计数器、字符计数器、线计数器和行计数器，分频关系如图3.6.1所示。

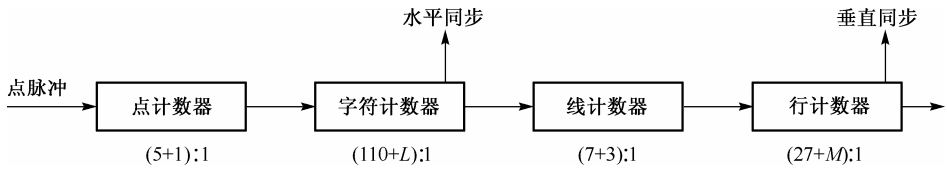


图 3.6.1　显示器的同步计数器

注：L 为水平回扫时间与屏幕右、右边缘部分消隐段的扫描时间折合的字符个数；M 为垂直回扫时间和屏幕上、下边缘部分消隐段的扫描时间折合的字符行数。

（8）该字符显示器每显示一个字符，应该访问基本显示缓存几次？应该访问字符发生器几次？

该字符显示器每显示字符的一行（即线）点，就应先访问一次基本显示缓存，接着访问一次字符发生器。故显示一个字符应该访问基本显示缓存 7 次，访问字符发生器 7 次。

（9）显示一行字符要访问基本显示缓存几次？访问字符发生器几次？

显示一行字符，应访问基本显示缓存共 7×110=770 次，访问字符发生器也是 770 次。

（10）在显示过程中，何时访问基本显示缓存？何时访问字符发生器？其访问地址由谁确定。

当点计数器循环计数一遍时，就访问一次基本显示缓存；访基本显示缓存所需的地址由行计数器和字符计数器的值决定。每读一次基本显示缓存，就紧跟着读一次字符发生器；字符发生器的访问地址由高位段和低位段构成，高位段由从基本显示缓存读出的字符编码提供，低位段由扫描时序电路中的线计数器的值充当。

6. 若 CRT 显示器采用逐行扫描的图形方式工作，其帧频为 50 Hz，分辨率为 1024×800 像素，能显示 65536 种颜色，求视频显示缓存 VRAM 的容量是多少？每一像素允许的读出（包括传至 CRT 管）时间是多少？

视频显示缓存 VRAM 的容量为分辨率乘以表示一个像素所需的存储量。每个像素允许有 65536 种颜色，故每个像素点需要 16 位二进制代码来表示它的颜色。于是，视频显示缓存 VRAM 的容量为：

$$\begin{aligned}1024\times800\times16(\text{bit}) &= 1024\times800\times2(\text{B})\\ &= 1638400(\text{B})\end{aligned}$$

每个像素允许的读出时间为

$$\frac{1}{50}\div(1024\times800)=0.02\div(819200)\approx24.4(\text{ns})$$

7. 若磁盘有两个记录面，磁盘的最内圈磁道直径是 2.5 英寸，最外圈磁道直径是 5 英寸；最大位密度 52000 bpi（位/英寸），磁道密度 1000tpi（道/英寸）；盘面分为 30 个扇区，每个扇区容量为 1024B；磁盘转速为 1200 rpm（转/分），设寻道时间为 10～40 ms。问：

（1）磁盘的总磁道数是多少？

记录面半径上记录区域的长度 l =(最外圈磁道直径 – 最内圈磁道直径)÷2
=(5 英寸 – 2.5 英寸)÷2
=1.25 英寸

每个记录面的磁道数 *n*（即每面道数）=道密度× l =1000× 1.25=1250（道/面）

磁盘的总磁道数 *N*=每面道数 *n*×2=2500（道）

（2）磁盘的非格式化容量是多少？

一条磁道的非格化式容量 i = 最内圈道周长×最大位密度
$= 2\pi r\times$ 最大位密度
$= 2\times3.14\times(2.5\div2)\times$ 最大位密度
$= 2.5\times3.14\times52000$
=408200(b)

=51025(B)

磁盘的非格式化容量 $I = iN = 51025 \times 2500\,(\text{B})$

$= 127562500\,(\text{B})$

$\approx 121.6(\text{MB})$

（3）磁盘的格式化容量是多少？

盘面分为 30 个扇区，是指每条磁道的扇区数为 30。

一条磁道的格式化容量 j = 每个扇区的容量×每道的扇区数 $= 1024 \times 30 = 30720(\text{B})$

磁盘的格式化容量= $j \times N = 30720 \times 2500 = 76800000(\text{B}) \approx 73(\text{MB})$

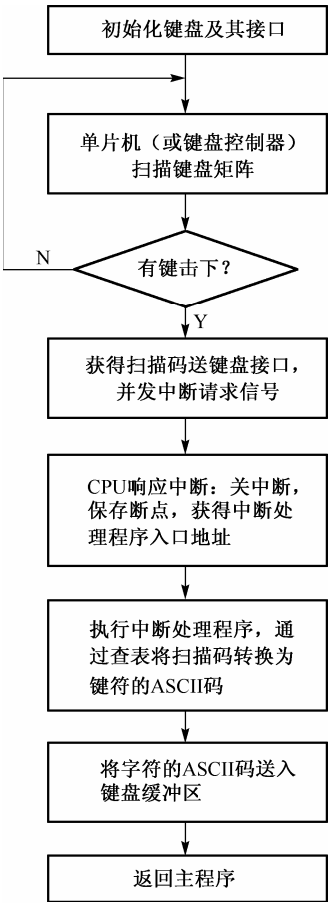


图 3.6.2　键盘中断方式的工作过程

（4）磁盘的平均寻道时间、平均旋转延迟时间和数据传输率各是多少？

平均寻道时间 $T_s = (10 + 40) \div 2 = 25(\text{ms})$

平均旋转延迟时间 T_r =磁盘旋转一圈的时间 ÷2

$= (60 \div 1200) \div 2$

$= 0.025\ (\text{s})$

$= 25(\text{ms})$

磁盘的数据传输率=(每道的非格式化容量 i×转速) ÷60

$= (51025 \times 1200) \div 60$

$= 1020500\ (\text{B/s})$

8．试用流程图描述键盘按中断方式工作的全过程。

键盘按中断方式工作的全过程可用图 3.6.2 所示的流程表示。

9．以打印机为例，阐述中断工作的全过程。

打印机以中断方式与 CPU 交换信息的全过程可归纳如下。

① CPU 执行指令送出命令字，以调用打印机进行初始化。完成初始化后，打印机置接口状态寄存器“忙”标志为 0，接口产生中断请求信号送中断控制器，后者发出中断请求信号 INTR 并送往 CPU。

② 若 CPU 允许响应中断，则在指令的最后一个时钟周期里向中断控制器发批准信号 INTA，在该指令结束时就转入中断周期 IT。

③ CPU 在 IT 周期依次进行关中断，保存断点（即断点压线），接收中断控制器发来的中断类型码，并转换为向量地址。以此地址访问中断向量表，从中取出打印中断处理程序入口地址，送入程序计数器 PC。

④ 转取指周期，执行中断处理地程序。在该程序中，CPU 从主存的打印缓冲区取一个

字符编码送往打印机的缓冲区，打印机将接口中的状态寄存器的 ACK 位置 1，CPU 据此继续输出下一个打印字符的编码，直至一行打印字符送为为止。

⑤ 打印机开始执行打印程序，且使“忙”标志为 1；同时 CPU 从堆线取回返回地址，返回主程序继续执行。

⑥ 当打印机印完一行字符后，置接口“忙”标志为 0，接口再次提出中断请求，请求 CPU 送下一行打印字符。

10．以读盘为例，阐述 DMA 方式工作的全过程。

以教材中的模型机磁盘调用为例，说明读盘过程。

① CPU 执行 DMA 初始化程序。主要完成预置 DMA 控制器的工作方式、传送方向（为接口→M）、主存缓冲区首址和批量传送的字节数；预置磁盘接口，即向接口送出读盘的寻址信息和读命令，并选择接口寄存器。

② CPU 启动磁盘进行寻道，等待起始扇区到达，开始连续地读盘操作，磁盘将读出的数据送入接口中的一个缓冲存储器。

③ 若缓冲存储器装满一个扇区容量的代码，则接口提示 DMA 控制器，后者提出总线请求 HRQ 送往 CPU。

④ 若 CPU 具备响应条件，则发出总线批准信号 HLDA，回送给 DAM 控制器。此后，CPU 与系统总线脱钩，DMA 控制器接管总线控制权。

⑤ DAM 控制器向接口送出批准信号 DACK、I/O 设备读命令 $\overline{\text{IOR}}$、存储器写命令 $\overline{\text{MEMW}}$ 和主存缓区首址。

⑥ 读出磁盘接口缓存 1 字节数据，写入主存缓冲区首地址，然后 DMA 控制器内的地址寄存器内容加 1，传送字节数减 1；重复进行，直至传送字节数减成 0 为止，表明批量传送完毕。DMA 控制器终止传送，将总线控制权交回 CPU。

⑦ DMA 批量传送结束，磁盘接口提出中断请求，经中断控制器发向 CPU，CPU 响应中断，执行磁盘中断处理程序。该程序调回磁盘接口状态信息，判断 DMA 传送有无错误或故障，有则进行出错处理，若无，则此次读盘成功。

11．试用流程图描述按 DMA 方式工作的写盘全过程。

按 DMA 方式工作的写盘全过程可用图 3.6.3 流程表示。

3.6.2 第 6 章习题解答与解题思路

1．简要解释下列名词术语

【答】

I/O 设备：计算机系统与人、其他设备或系统之间进行信息交换的装置，主机以外的大部硬件设备都称为外围设备或 I/O 设备，如打印机、显示器和磁盘等设备。

并行传送：同时采用多根传送线，并行地传送一字节或一个字，称为并行传送。

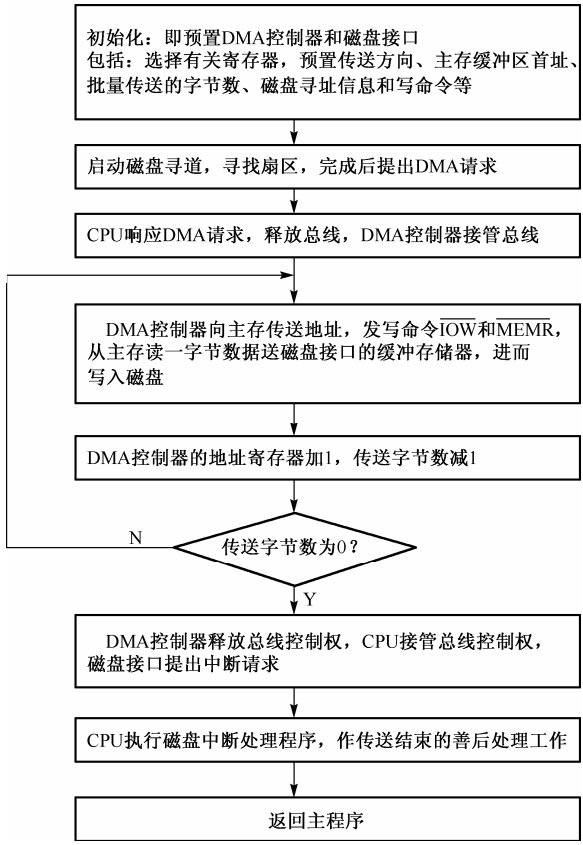


图 3.6.3　DMA 方式的写盘过程

串行传送：采用单根信号传送线（对公共地形成电位差）或用一对传送线（一根信号线，一根地线）逐位串行地传送数据代码，称为串行传送。

扫描码：通过硬件（或软件）扫描，识别所按下的键的行列位置，表示该行列位置的代码称为扫描码，也称为位置码或座标码。

逐行扫描法：将键连成 m 行×n 列的矩阵，在执行键盘扫描程序时，CPU 从第 0 行开始，逐行为 0（其余各行为 1）的方式，将代码 0 送往行线，并取回列线状态来判别按键位置，这种确定按键位置的方法称为逐行扫描法。

行列扫描法：先逐列为 1 地步进扫描，CPU 测试是哪一列为 1 时行线组输出为“1”，从而判明按键的列号；再逐行为“1”地步进扫描，CPU 测试哪一行为 1 时，列线组输出也是“1”，即判明哪行按了键。CPU 根据行、列扫描结果便能确定按键的位置。行列号形成对应的扫描码（或位置码），这种键盘扫描法称为行列扫描法。

像点（像素）：将显示屏幕沿水平和垂直方向划分成 $m \times n$ 个最少的显示区域，这个最小区域就是显示器的最小显示单元，对应屏上一个显示点，该最小显示区域称为像点或像素。

分辨率：指显示器一屏所能表示的像素的最大个数。

灰度级：在黑白显示器中，灰度级是指所显示的像素点的亮暗程度，在彩色显示器中，则表示颜色的不同。

字符/数字方式：对显示器（或打印机）而言，以字符作为显示（或打印）内容的基本单元，这种方式称为字符/数字方式，也称为文本显示（或打印）方式。

图形方式：以像点作为显示内容的基本单元的显示方式称为圆形方式。某些打印机也有相似的圆形打印方式。

位密度：沿磁道圆周，单位距离可记录的二进制代码位数称为位密度。

道容量：指一条磁道所能存放的二进制代码位数。

2．简答题

（1）磁盘的工作速度是由什么因素决定的？

（2）举例说明一种实用的键码形成方法。

（3）简述 CRT 显示器由字符代码到字符点阵的转换过程。

（4）简述激光打印的基本原理。

（5）字符显示器为了实现同步控制，一般应设置哪几级同步计数器？

（6）图形显示器应设置哪几级同步计数器？

【答】

（1）磁盘的工作速度是由三部分组成的，即平均寻道时间、平均旋转延迟时间和数据传输率。前两部分的决定因素主要是磁盘主轴转速和磁头移动速度，数据传输率则与磁盘读/写速度、主存速度和接口逻辑线路有关。

（2）设有 4×4 的扫描式键盘矩阵如图3.6.4所示，用逐行扫描法来获取键码。

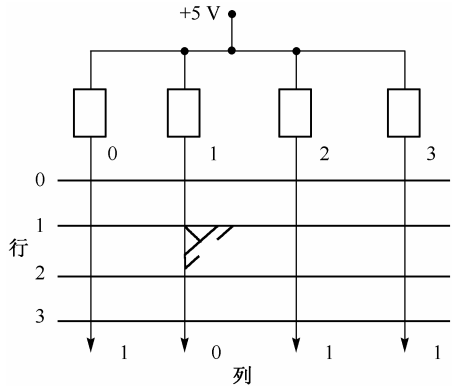


图 3.6.4　4×4 扫描式键盘矩阵

键码形成过程如下：

① 当 5 号键按下时，产生中断请求，CPU 执行键盘扫描程序。

② 在扫描程序中，CPU 使第 0 行线为 0（其余行为 1），同时取出各列线状态，由于 0 行上无键按下，故各列均为 1。

③ CPU 接着使第 1 行线为 0，由于 5 号键已按下，于是 CPU 从取出的列线状态中发现第 1 列为 0，于是得到按键扫描码（1、1）。

④ 根据扫描码（1、1）查找键码转换表，即可获得 5 号键的键码。

（3）在字符方式下，首先从显示缓存取出字符编码；将字符编码作为字符发生器地址的高位部分，将 CRT 控制器提供的扫描时序（即线计数器输出）作为地址低位部分送字符发生器；读取出该字符的一行点阵代码，并行送入移位寄存器；经并-串转换，送出点脉冲信号到 CRT 管，即可在屏上显示该字符的一行（即线）点阵图案。

（4）激光打印机利用激光扫描技术将经过调制的、载有字符点阵信息或图形信息的激光束扫描在光导材料上，并利用电子摄影技术让激光照射过的部分曝光，形成图形的静电潜像。再经过墨粉显影、电场转印和热压定影，便在纸上印刷出可见的字符或图形。

（5）字符显示器一般应设四级同步计数器，依次为点计数器、字符计数器、线计数器和行计数器。

（6）一般应设三级同步计数器，依次为点计数器、字节计数器和线计数器。

3. 某 CRT 显示器作为字符显示，能显示 64 种字符，每帧可显示的最大容量为 25 行×64 列字符，每个字符采用 7×8 点阵，即横向 7 点、纵向 8 点，则字符发生器的容量为多少？

【答】

每个字符都是采用 7×8 点阵规格来显示，习惯上常将横向 7 点的代码放一字节单元（最高位为 0），一个字符有 8 行点，故一个字符需要 8 字节来放它的点阵代码。64 种字符则需要 8×64=512 B，故该字符发生器容量为 512 B。理论上，字符发生器容量为

$$7\times8\times64=3584(\text{b})$$

4．某 CRT 显示器作为字符显示，每帧可以显示 25 行×80 列字符，每个字符采用横 7×纵 9 点阵，字符间横向间距 2 点，行间间距 5 点。则点计数器应如何分频？字符计数器应如何分频？线计数器应如何分频？行计数器应如何分频？

【答】

点计数器分频关系为$(7+2)\text{∶}1$。

字符计数器分频关系为$(80+L)\text{∶}1$。

线计数器分频关系为$(9+5)\text{∶}1$。

行计数器分频关系为$(25+M)\text{∶}1$。

注：L 为屏幕两边非线性失真和水平回扫消隐部分折合的字符个数，M 为屏幕顶部、底部非线性失真和垂直回扫消隐部份折合的字符行数。

5．某图形显示器的分辨率为横向点 800×纵向点 600，则同步计数器的点计数器应如何分频？字节计数器应如何分频？线计数器应如何分频？

【答】

点计数器分频为$8\text{∶}1$。

字节计数器分频为$\left(\dfrac{800}{8}+L\right):1$。

线计数器分频为$(600+M):1$。

注：L 为屏幕两边非线性部分和水平回扫等消隐段折合成的字节数，M 是垂直回扫和屏幕上下边缘非线性失真的消隐段折合的数线。

6．在字符显示器与点阵针式打印机中都有字符发生器，它们的主要区别是什么？

【答】

字符显示器中的字符发生器是按点阵行进行读出，即每次读取字符点阵的一行代码，故字符的点阵代码在字符发生器中是按行存储。点阵针式打印机的字符发生器是将字符的点阵代码按列存放，每次读取字符点阵的一列代码，进行打印。

7．若要将图形显示的分辨率从 800×600 提高到 1024×1024，在适配卡上应采用哪些措施？

【答】

应采取的主要措施是：增加刷新存储器（即显示缓存）容量，以便存放一帧像点代码；提高点脉冲和锯齿波频率，缩短显示缓存的存取周期；重新设置 CRTC 中相应寄存器的初值和同步计数器分频关系等。

8．若要将显示字符从 7×9 点阵放大为 14×18 点阵，请提出一种实现方案。

【答】

扫描显示第一帧时，显示缓存中存放的是字符 7×9 点阵代码。扫描显示第二帧时，将显示缓存的内容修改为字符 14×18 点阵代码，即可实现放大。

9．若要将显示画面（字符型）自下而上地滚动，请提出一种实现方案。

【答】

显示不同帧时，自下而上地改变显示画面中字符的编码在显示缓存中的存放位置，即可实现相应滚动。

10．若要将显示字符 A 从屏幕左上角逐渐地移向屏幕右下角，请提出一种实现方案。

【答】

显示不同帧时，改变每帧 A 编码在显示缓存中的存放位置，即可实现相应移动。

11．若要让一个图形在屏幕上旋转，请提出一种实现方案。

【答】

显示不同帧时，按旋转方向改变图形在显示缓存中的存放位置，即可实现图形的旋转。

第 4 章　考 研 辅 导

“计算机组成原理”作为计算机专业非常重要的专业基础课，是很多高校的考研科目或者复试科目。从 2009 年开始，该课程已作为计算机专业研究生考试全国统考科目，其考查目标有三点：

① 理解单处理器计算机系统中各部件的内部工作原理、组成结构及相互连接方式，具有完整的计算机系统的整机概念。

② 理解计算机系统层次化结构概念，熟悉硬件与软件之间的界面，掌握指令集体系结构的基本知识和基本实现方法。

③ 能够运用计算机组成的基本原理和基本方法，对有关计算机硬件系统中的理论和实际问题进行计算、分析，并能对一些基本部件进行简单设计。

本章根据统考考纲要求，结合 2009 年、2010 年部分统考考题，对“计算机组成原理”课程的主要内容进行复习，并给出几份考研真题的解答，供读者参考。

4.1　计算机组成原理考研复习

复习内容包括 5 方面：数据的表示和运算，存储子系统，CPU 子系统，总线，以及输入/输出子系统。

4.1.1　数据的表示和运算

1．数据的表示

数据分为数值型和非数值型两大类，数值型数据的表示包含三个因素：数据的大小、数据的符号和数据的小数点。数据的大小用二进制、八进制、十六进制、二-十进制等表示；数据的符号用原码、补码、反码等表示；小数点用定点或浮点表示。

下面是 2009 年统考中涉及补码表示的一道考题。

【例 4-1】 一个 C 语言程序在一台 32 位机器上运行。程序中定义了三个变量 x、y 和 z，其中 x 和 z 为 int 型，y 为 short 型。当 x = 127，y = –9 时，执行赋值语句“z=x+y;”后，x、y 和 z 的值分别是（　　）。

A．x=0000007FH，y=FFF9H，z=00000076H

B．x=0000007FH，y=FFF9H，z=FFFF0076H

C．x=0000007FH，y=FFF7H，z=FFFF0076H

D．x=0000007FH，y=FFF7H，z=00000076H

答案：D。

分析该题，在四个备选答案中，A、B 的 y 不是-9 的补码表示，应排除；C 中的运算结果 z 也不是正数的补码表示，所以正确答案是 D。

关于定点数和浮点数，需要强调它们的表现形式和表示范围。

（1）定点表示法

定点数包含无符号数、定点整数、定点小数三种类型，它们的表示范围如下（以 8 位数为例）：

无符号数　　00000000 ～ 11111111
　　（0）　　（255）

定点整数
$11111111_{原}$ ～ $01111111_{原}$
（–127）　　（127）
$10000000_{补}$ ～ $01111111_{补}$
（–128）　　（127）

定点小数
$1.1111111_{原}$ ～ $0.1111111_{原}$
$-(1-2^{-7})$　　$(1-2^{-7})$
$1.0000000_{补}$ ～ $0.1111111_{补}$
（–1）　　$(1-2^{-7})$

注意，在原码表示中，由于包含了+0 和–0，所以上例的最小定点整数是–127，最小定点小数是$-(1-2^{-7})$。而在补码表示中没有–0，因而可以用多余的代码组合来表示最小定点整数–128，以及最小定点小数–1。

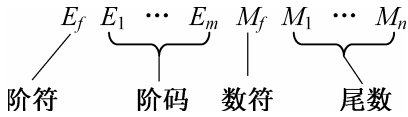
（2）浮点表示法

1）格式

浮点数真值：

$$N = \pm R^{E} \times M$$

浮点数机器格式：



R：阶码底，隐含约定。

E：阶码，为定点整数，补码或移码表示，其位数决定数值范围；阶符表示数的大小。

M：尾数，为定点小数，原码或补码表示，其位数决定数的精度；数符表示数的正负。尾数常采用规格化形式，即 $1/2 \leqslant |M| < 1$，以使尾数的各位都是有效数位。

2）表示范围与精度

浮点数的表示范围在最小浮点数和最大浮点数之间，浮点数的精度即是最小浮点正数。阶符 1 位，阶码 m 位，补码表示，以 2 为底；数符 1 位，尾数 n 位，补码表示，规格化，则

最小浮点数：
阶码为最大数：$2^{m}-1$
尾数为绝对值最大的负数：–1

最大浮点数：
阶码为最大数：$2^{m}-1$
尾数为最大数：$1-2^{-n}$

最小浮点正数：$\begin{cases}\text{阶码为最小数：}-2^{m}\\ \text{尾数为最小正数：}2^{-1}\end{cases}$

【例 4-2】 某规格化浮点数用补码表示，其中阶码 6 位，含 1 位阶符；尾数 10 位，含 1 位数符。

表示范围：$-2^{31}\sim 2^{31}$

表示精度：2^{-33}

（3）IEEE754 标准

在当前主流微机中广泛采用 IEEE754 标准，包含了三种浮点格式：短实数、长实数和临时实数，它们的格式如下：

	符号位	阶码	尾数	总位数	数值
短实数	1	8	23	32	$(-1)^{s}\times 1.M\times 2^{e-127}$
长实数	1	11	52	64	$(-1)^{s}\times 1.M\times 2^{e-1023}$
临时实数	1	15	64	80	

其中，*s* 为 0 表示正数，*s* 为 1 表示负数；短实数 *e* 为 1～254（8 位），*e*=真值+127；*M* 为 23 位，其中的“1.”隐藏；长实数 *e* 为 1～2046（11 位），*e* = 真值+1023；*M* 为 52 位。

【例 4-3】 十进制数 5 的短实数 IEEE754 代码为（　　）。

A．01000000101000000000000000000000

B．11000000101000000000000000000000

C．01100000101000000000000000000000

D．11000000101100000000000000000000

本题中，先将 5 转换为二进制代码，再表示为浮点真值形式 1.01×2^{2}，然后按短实数格式计算：符号位为 0，阶码=2+127=129，尾数“1.”隐藏，所以正确答案是 A。

2. 定点数的运算

定点数的主要运算包括加、减、乘、除四则运算，每种运算又分为原码运算和补码运算。关于这部分内容，需要掌握原码运算和补码运算的区别、四则运算方法，以及溢出判断方法。

原码运算：操作数和结果均用原码表示，绝对值参加运算，符号单独处理。

补码运算：操作数和结果均用补码表示，符号参加运算。

（1）补码加减运算

1）算法

加法：两数直接相加。

减法：减数变补后与被减数相加。

2）溢出判断方法

① 单符号法

设 S_A、S_B 分别表示数 *A*、数 *B* 的符号，S_f 为结果的符号，则有

$$\text{溢出}=\overline{S}_A\overline{S}_B S_f + S_A S_B \overline{S}_f$$

② 双符号法

设 S_{f1}、S_{f2} 分别表示第一符号位、第二符号位，则有

$$溢出=S_{f1} \oplus S_{f2}$$

③ 进位判断法

设 C_f、C 分别表示符号位的进位、最高有效位的进位，则有

$$溢出=C_f \oplus C$$

（2）乘法运算

基本的乘法运算包括原码一位乘法和补码一位乘法。

1）原码乘法（一位乘）

运算规则：用乘数末位作判断位，根据该位状态决定 $A+B$ 或 $A+0$。（A 为部分积累加和，B 为被乘数）

2）补码乘法（一位乘）

运算规则：用乘数末位和其后的附加位作判断位，根据该两位状态决定 $A+B$、$A-B$ 或 $A+0$。

（3）除法运算

基本的除法运算包括原码不恢复余数法和补码不恢复余数法。

原码除法（不恢复余数）的运算规则为：根据余数的正负决定商值，根据商值决定下步操作。

补码除法（不恢复余数）的运算规则为：根据余数和除数的符号决定商值，根据商值决定下步操作。

3. 浮点数的运算

主要掌握浮点加减运算过程，其中“对阶”和“规格化”是两个重要的基本概念。

对阶：小阶向大阶对齐。小阶增大，尾数右移。

尾数加/减：尾数通常用补码表示，按补码加减运算规则进行加/减。

结果规格化：根据尾数绝对值的情况，分为左移规格化（左规）和右移规格化（右规）两种方式。

左规：尾数绝对值<1/2 时，尾数左移，阶码减 1。

右规：尾数绝对值≥1 时，尾数右移，阶码加 1。

【例 4-4】 浮点数加、减运算过程一般包括对阶、尾数运算、规格化、舍入和判溢出等步骤。设浮点数的阶码和尾数均采用补码表示，且位数分别为 5 位和 7 位（均含 2 位符号位）。若有两个数 $X=27\times29/32$，$Y=25\times5/8$，则用浮点加法计算 $X+Y$ 的最终结果是（　　）。

A. 00111　1100010

B. 00111　0100010

C. 01000　0010001

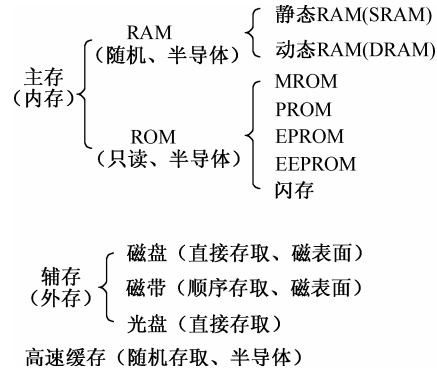
D. 发生溢出

数 X 的浮点格式为 00111　00.11101，数 Y 的浮点格式为 00101　00.10100，对阶后，数 Y 的浮点格式为 00111 00.00101；两尾数相加，结果为 01.00010；绝对值大于 1，需右规，即尾数右移为 00.10001，阶码加 1 为 01000，阶码发生了上溢。所以最终结果是发生了溢出。

答案：D。

4.1.2 存储子系统

1．存储器的分类



2．随机存储器

（1）SRAM

存储原理：利用触发器原理存储信息。

特点：速度快，非破坏性读出，信息易失；电源正常时能长期保存信息，不需刷新；常用作高速缓存。

（2）DRAM

存储原理：利用电容存储电荷原理存储信息。

特点：速度快，非破坏性读出，信息易失。平时无电源供电，需动态刷新。常用作主存。

【例 4-5】 下列有关 RAM 和 ROM 叙述中，正确的是（　　）。

Ⅰ．RAM 是易失性存储器，ROM 是非易失性存储器

Ⅱ．RAM 和 ROM 都采用随机存取方式进行信息访问

Ⅲ．RAM 和 ROM 都可用作 Cache

Ⅳ．RAM 和 ROM 都需要进行刷新

A．仅Ⅰ和Ⅱ　　B．仅Ⅱ和Ⅲ　　C．仅Ⅰ、Ⅱ和Ⅳ　　D．仅Ⅱ、Ⅲ和Ⅳ

注意：尽管 ROM 是只读存储器，但采用的访问方式是随机方式，即可按地址访问 ROM 的任一单元，访问时间与单元所在的位置无关。

所以，答案是 A。

（3）动态刷新

定义：按所存信息定期向电容补充电荷。

方式：按行读。

刷新周期安排方式，如图 4.1 所示。

（4）主存与 CPU 的连接

① 扩展存储容量

位扩展：扩展存储单元的位数。将多片存储芯片的地址、片选、读写端并联，数据端分开。

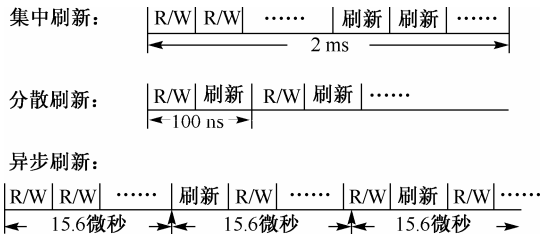


图 4.1　三种刷新周期安排方式

字扩展：扩展存储单元的数量，将多片存储芯片的地址、数据、读写端并联，片选端分开。

② 连接 CPU 芯片

地址线的连接：通常将存储芯片的地址线（其位数由存储芯片容量决定）与 CPU 地址线的低位相连，CPU 地址线的高位用于形成片选信号。

数据线的连接：当存储芯片数据线与 CPU 数据线数量不等时，扩展存储芯片位数，并与 CPU 数据线相连。

读/写线的连接：存储芯片的读/写端直接与 CPU 读/写线相连。

片选线的连接：存储芯片的片选端与片选逻辑电路的输出相连。片选逻辑电路由门电路或译码器组成，用 CPU 地址高位和访存控制信号驱动。

③ 存储器的设计

【例 4-6】 某计算机主存容量为 64 KB，其中 ROM 区为 4 KB，其余为 RAM 区，按字节编址。现要用 2K×8 位的 ROM 芯片和 4K×4 位的 RAM 芯片来设计该存储器，则需要上述规格的 ROM 芯片数和 RAM 芯片数分别是（　　）。

A．1、15　　B．2、15　　C．1、30　　D．2、30

答案：D

【例 4-7】 假定用若干个 2K×4 位的芯片组成一个 8K×8 位的存储器，则地址 0B1FH 所在芯片的最小地址是（　　）。

A．0000H　　B．0600H　　C．0700H　　D．0800H

答案：D。

3．高速缓冲存储器

（1）Cache 的基本工作原理

利用程序访问的局部性原理，将 CPU 在较短时间内频繁访问的程序和数据从主存拷贝到 Cache 中，CPU 高速访问 Cache，以提高访存速度。

需解决的问题包括：

① 地址映射：包括三种方式——直接映射、全相联映射、组相联映射。

② 访问命中：命中率的计算公式如下

$$\text{命中率}=\frac{\text{访问合中次数}}{\text{访问次数}}\times 100\%$$

③ Cache 内容替换。

④ 数据一致性。

（2）地址映射

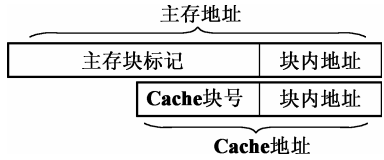
注意数据/代码是否在 Cache 的判断：通过判断该数据/代码的地址是否在 Cache 中，而不是判断该数据本身。

将主存与 Cache 划分为若干大小相同的块（页）。将主存的部分块放入 Cache 块中。

Cache 分为两部分：地址部分（标记）和数据部分。

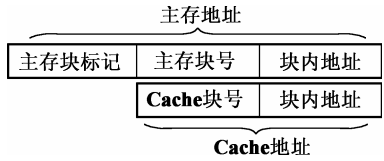
① 全相联映射：主存的每一块可以映射到 Cache 的任一块。

地址结构：



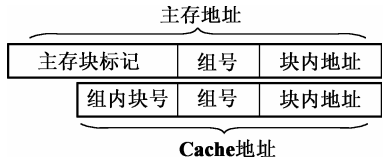
② 直接映射：主存的每一块只能映射到 Cache 的固定块中。

地址结构：



③ 组相联映射：主存的每一块可以映射到 Cache 的多个固定块。

地址结构：



【例 4-8】 某计算机的 Cache 共有 16 块，采用 2 路组相联映射方式（即每组 2 块）。每个主存块大小为 32 字节，按字节编址。主存 129 号单元所在主存块应装入到的 Cache 组号是（　　）。

A. 0　　B. 2　　C. 4　　D. 6

答案：C

【例 4-9】 假设某计算机的存储系统由 Cache 和主存组成。某程序执行过程中访存 1000 次，其中访问 Cache 缺失（未命中）50 次，则 Cache 的命中率是（　　）。

A. 5%　　B. 9.5%　　C. 50%　　D. 95%

答案：D

（3）替换算法

替换算法包括 FIFO、LRU 等。

（4）读/写操作

读操作：将 Cache 标记与主存地址中的主存块标记比较，若符合，则访问命中，从 Cache 读取数据；若不符，则访问失败，从主存读取数据。

写操作：包括直写和回写。

直写：写入 Cache 时也同时写入主存。

回写：写入 Cache 时暂不写入主存，直到该内容将被替换时才写入主存。

4．虚拟存储器

为扩大用户编程空间，将主存和辅存的空间统一编址，形成虚拟存储空间。

（1）页式虚拟存储器

虚存空间和主存空间均分成若干大小相同的页。虚-实地址的转换通过页表实现，见图 4.2。

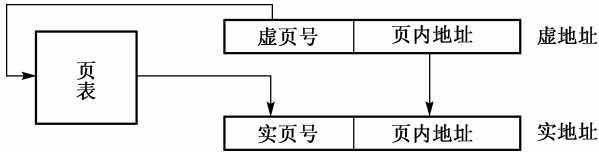


图 4.2　页式虚拟存储器的地址转换

（2）段式虚拟存储器

虚存空间按程序模块分成若干大小不等的段。虚-实地址的转换通过段表实现，见图 4.3。

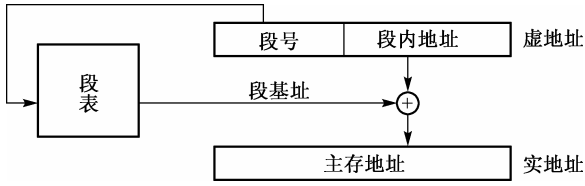


图 4.3　段式虚拟存储器的地址转换

（3）段页式虚拟存储器

虚存空间按程序模块分成若干大小不等的段，每段分为若干大小相同的页。虚-实地址的转换通过段表和页表实现，见图 4.4。

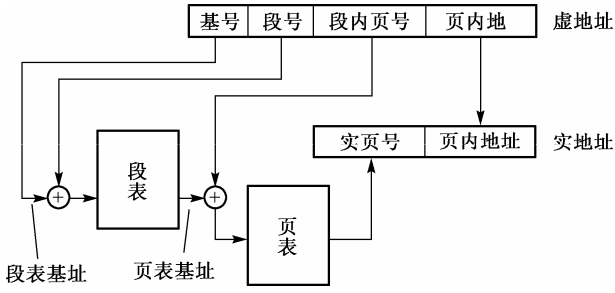


图 4.4　段页式虚拟存储器的地址转换

4.1.3 CPU子系统

1．指令的寻址方式

（1）指令寻址

顺序寻址：通过 PC+1 实现。

跳跃寻址：通过转移类指令实现。

（2）数据寻址

数据寻址包括：立即寻址、直接寻址、隐含寻址、间接寻址、寄存器寻址、寄存器间址、基址寻址、变址寻址、相对寻址、堆栈寻址

基址和变址的区别：基址主要用于存储空间的分配，基址寄存器的内容通常由操作系统确定；变址主要用于数组的处理，变址寄存器的内容通常由用户设定。

【例 4-10】 某机器字长 16 位，主存按字节编址，转移指令采用相对寻址，由两个字节组成，第一字节为操作码字段，第二字节为相对位移量字段。假定取指令时，每取一个字节 PC 自动加 1。若某转移指令所在主存地址为 2000H，相对位移量字段的内容为 06H，则该转移指令成功转移后的目标地址是（　　）。

A．2006H　　B．2007H　　C．2008H　　D．2009H

答案：C。

2．CISC和RISC

（1）CISC 的特点

① 指令数量多。

② 寻址方式多，指令格式多，指令字长不固定。

③ 可访存指令不受限制。

④ 各种指令使用频率相差大。

⑤ 各种指令执行时间相差大。

⑥ 大多采用微程序控制。

（2）研究和结论

研究：软件中大部分指令为简单指令；软件中的简单指令约占总运行时间的 20%；造成控制电路复杂的主要原因是由于复杂指令的存在。

结论：从指令集中去掉复杂指令，复杂指令功能由软件实现，可简化电路设计；去掉微程序，采用硬连控制方法，提高处理器速度。

（3）RISC 的特点

① 简单固定的指令格式。

② 减少寻址方式和指令数量。

③ 采用流水线技术。

④ 采用大量寄存器。

⑤ 采用硬连控制。

⑥ 采用专门的访存指令访存。

⑦ 采用优化编译技术。

【例 4-11】 下列关于 RISC 的叙述中，错误的是（　　）。

A. RISC 普遍采用微程序控制

B. RISC 大多数指令在一个时钟周期内完成

C. RISC 的内部通用寄存器数量相对 CISC 多

D. RISC 的指令数、寻址方式和指令格式种类相对 CISC 少

答案：A。

3．CPU的功能和组成

（1）功能

① 执行指令序列：取指令，分析指令，执行指令。

② 操作控制：提供执行指令所需的各种微命令（微操作信号）。

③ 时间控制：对各种操作进行时间上的控制。

④ 数据加工：对数据进行算术运算和逻辑运算处理。

（2）组成

① 控制器：由 PC、IR、MAR、MDR、PSW、指令译码器、时序发生器、微操作信号发生器等组成。

② 运算器：由 ALU、输入暂存器、输出移位器、通用寄存器组等组成。

（3）总线结构

CPU 内部各寄存器之间、寄存器与运算部件之间通过内部总线连接；CPU 与主存及 I/O 接口之间通过系统总线连接。

4．指令执行过程

（1）指令执行的周期

① 取指周期：PC 给出指令地址，取指后 PC 内容递增；转移时，指令地址在执行周期被修改。

② 取数周期：根据寻址方式计算操作数地址并取出操作数。

③ 执行周期：按指令操作码完成相应操作，传送结果并记录状态信息。

（2）指令周期与机器周期和时钟周期的关系

① 指令周期：一条指令从取出到执行完成所需要的时间。

② 机器周期：指令周期划分为几个阶段，每个阶段所需的时间，也称为 CPU 工作周期（通常等于访存时间）。

③ 时钟周期：时钟频率的倒数，也称为节拍或 T 周期（通常等于 CPU 一步操作时间）。

一个指令周期包含若干机器周期，一个机器周期包含若干时钟周期。

（3）指令的执行

① 执行过程：取指令 → 取操作数（若无操作数，直接进入执行；若需操作数，则用一个取数周期；若需两个操作数，则用两个取数周期）→ 执行操作。

② 指令之间的衔接方式：包括串行的顺序执行方式、并行的重叠处理方式、流水执行方式、指令执行的操作流程与微操作序列。

③ 指令操作流程：将一条指令的执行过程分解为一组功能部件（寄存器）级的操作序列（常以流程图形式描述）。

④ 微操作序列：将功能部件级的操作序列进一步分解为一组最基本的不可再分割的微操作序列（常以微命令序列描述）。

（4）数据通路结构

CPU 数据通路阶梯如图4.5所示，i 表示某寄存器输入控制信号，o 表示某寄存器输出控制信号。

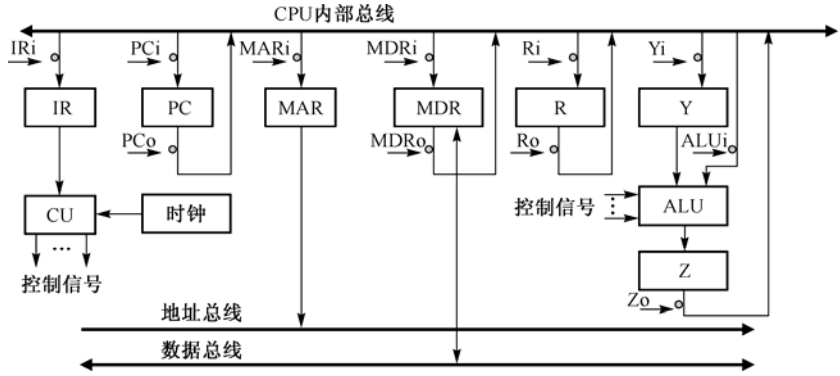


图 4.5　CPU 数据通路结构

（5）指令流程

【例 4-12】 某计算机字长 16 位，采用 16 位定长指令字结构，部分数据通路结构如图 4.6 所示。

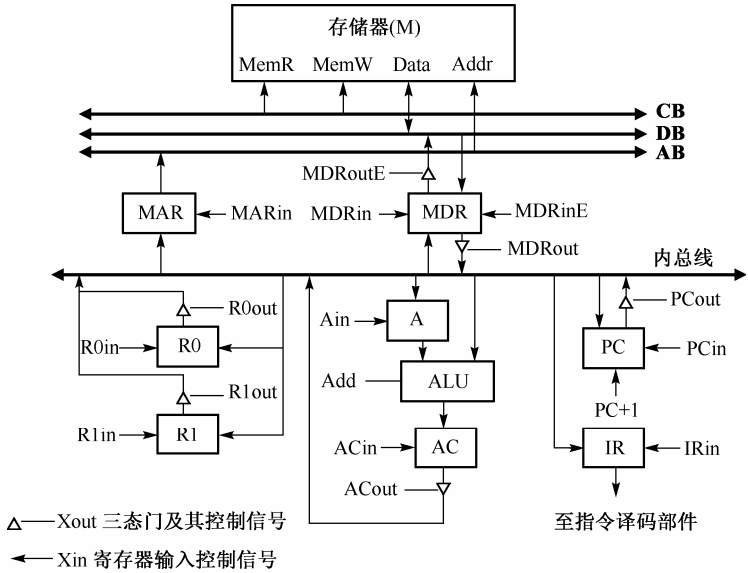


图 4.6　某机部分数据通路结构

图 4.6 中，所有控制信号为 1 时表示有效、为 0 时表示无效，如控制信号 MDRinE 为 1 表示允许数据从 DB 打入 MDR，MDRin 为 1 表示允许数据从内总线打入 MDR。假设 MAR 的输出一直处于使能状态。加法指令“ADD (R1), R0”的功能为 (R0)+ ((R1))→(R1)，即将 R0 中的数据与 R1 的内容所指主存单元的数据相加，并将结果送入 R1 的内容所指主存单元中保存。

表 4-1 给出了上述指令取指和译码阶段每个节拍（时钟周期）的功能和有效控制信号，请按表中描述方式用表格列出指令执行阶段每个节拍的功能和有效控制信号。

答案：如表 4-2 所示。

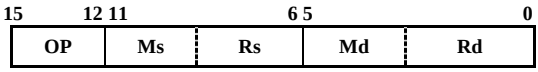
表 4-1 取指和译码阶段每个节拍的功能和有效控制信号

时钟	功能	有效控制信号
C1	MAR←(PC)	PCout，MARin
C2	MDR←M(MAR) PC←(PC)+1	MemR，MDRinE PC+1
C3	IR←(MDR)	MDRout，IRin
C4	指令译码	无

表 4-2 指令执行阶段每个节拍的功能和有效控制信号

时钟	功能	有效控制信号
C5	MAR←(R1)	R1out，MARin
C6	MDR←M(MAR)	MemR，MDRinE
C7	A←(MDR)	MDRout，Ain
C8	AC←(R0)+(A)	R0out，Add，Acin
C9	MDR←(AC)	ACout，MDRin
C10	M(MAR)←(MDR)	MDRoutE，MemW

【例 4-13】 某计算机字长为 16 位，主存地址空间大小为 128 KB，按字编址。采用单字长指令格式，指令各字段定义如下：



转移指令采用相对寻址，相对偏移量用补码表示。寻址方式定义如表 4-3 所示。

表 4-3 寻址方式定义

Ms/Md	寻址方式	助记符	含 义
000B	寄存器直接	Rn	操作数=(Rn)
001B	寄存器间接	(Rn)	操作数=((Rn))
010B	寄存器间接、自增	(Rn)+	操作数=((Rn)) (Rn)+1→Rn
011B	相对	D(Rn)	转移目标地址=(PC)+ (Rn)

请回答以下问题：

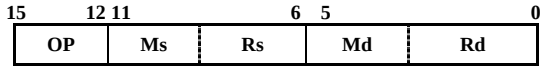
（1）该指令系统最多可有多少条指令，该计算机最多有多少个通用寄存器，存储器地址寄存器(MAR)和存储器数据寄存器至少各需要多少位（MDR）？

（2）转移的目标地址范围是多少？

（3）若 0010B 表示加法操作（助记符 add)），寄存器 R4 和 R5 的编号分别为 100B 和 101B，R4 的内容为 1234H，R5 的内容为 5678H，地址 1234H 中的内容为 5678H，地址 5678H 中的内容为 1234H，则汇编指令“add (R4), (R5)+”（逗号前为原操作数，逗号后为目标操作数）对应的机器码是什么（用十六进制表示）？该指令执行后，哪些寄存器和存储单元的内容会改变？改变后的内容是什么？

【答】

（1）根据指令格式

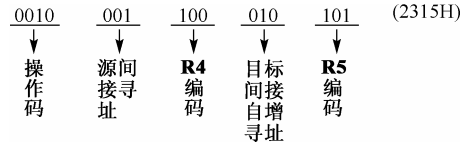


因为操作码为 4 位，所以最多有 16 条指令。因为用来表示寄存器的位数是 3 位，所以最多有 8 个通用寄存器。

存储空间按字编址，因此存储单元数量为 128 KB/2=64 KB。需要 16 位地址，所以地址寄存器 MAR 为 16 为。因为字长 16 位，所以数据寄存器 MDR 为 16 位。

（2）因为地址位数和字长都是 16 位，所以 PC 和通用寄存器的位数均为 16 位，转移目标地址位数为 16 位，从而转移目标地址范围为 0000H～FFFFH。

（3）指令“add (R4), (R5)+”操作码为 0010，源操作数采用间接寻址，编码 001，目标操作数采用间接自增寻址，编码 010，R4 的编码为 100，R5 的编码为 101B，所以对应的机器码为



指令执行后，结果存入 5678H 单元，因此由原来的 1234H 变为 68ACH，R5 的内容加 1 后变为 5679H。

（6）时序控制方式

1）同步控制方式

① 定义：各项操作受统一时序信号控制。

② 特点：有明显时序时间划分；时钟周期（节拍）时间固定；各步操作的衔接、各部件之间的数据传送受严格同步定时控制。

③ 应用场合：用于 CPU 内部、设备内部、总线操作（各挂接部件速度差异小、传送时间确定、传送距离较近）。

④ 三种方案：完全统一的机器周期和节拍；不同节拍的机器周期；中央控制和局部控制相结合。

2）异步控制方式

① 定义：各项操作按需要安排不同时间，不受统一时序约束。

② 特点：无严格时钟周期划分；各操作间的衔接、各部件之间的数据传送采取异步应答方式。

③ 应用场合：用于异步总线操作（各挂接部件速度差异大、传送时间不确定、传送距离较远）。

3）联合控制方式

联合控制方式是指同步控制和异步控制相结合。

（7）控制器工作原理

1）组合逻辑控制器

综合化简产生微命令的条件，形成相应逻辑式，用组合逻辑电路实现。执行指令时，由组合逻辑电路（微命令发生器）在相应时间发出所需微命令，控制有关操作。

优点：速度快。缺点：设计不规整，结构零乱，不易修改、扩充指令系统功能。

2）微程序控制器

其控制过程包括：将微命令以代码形式编成微指令，控制一步操作；若干微指令编成一段微程序，解释执行一条机器指令；微程序事先存放在一个控制存储器中，执行机器指令时再取出。

优点：结构规整，设计效率高，性价比高，可靠性高，易于修改、扩展指令系统功能。

缺点：速度较慢，执行效率受影响。

① 基本概念

微程序：包含若干微指令，解释执行一条机器指令。

微指令：产生微命令，控制完成机器指令功能的一步操作。

微命令：控制完成微操作的命令。

② 微指令编码方法：包括直接控制法、分段直接编译法、分段间接编译法、混合控制法等。

③ 微地址形成方式包括：微程序入口地址的形成，由机器指令操作码转换；后继微地址的形成，是增量方式、断定方式及这两种方式的结合。

（8）指令流水线

① 指令流水线（如图4.8所示）：将一条指令的执行过程划分为若干阶段，用不同部件分别完成各阶段相应操作，将多条指令的不同阶段用不同部件同时完成相应操作，每个时钟执行一条指令。

图 4.8　指令流水线

② 超标量流水线（如图4.9所示）：设置多套部件，用多个相同部件同时完成多条指令相同阶段的操作，每个时钟执行多条（两条）指令。

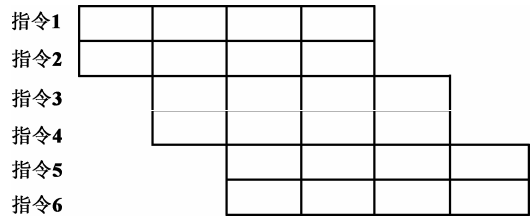


图 4.9　超标量流水线

③ 影响流水线的主要因素：数据相关，程序相关。

【例 4-14】 某计算机的指令流水线由四个功能段组成，指令流经各功能段的时间（忽略各功能段之间的缓存时间）分别为 90 ns、80 ns、70 ns 和 60 ns，则该计算机的 CPU 时钟周期至少是（　　）。

A. 90 ns　　B. 80 ns　　C. 70 ns　　D. 60 ns

答案：A。

4.1.4　总线

1．总线概述

（1）总线的分类

① 片内总线：芯片内部的总线。

② 系统总线：CPU、主存、I/O 之间的总线。分为数据总线、地址总线、控制总线。

③ 通信总线：计算机系统之间或与其他系统之间的总线，可分为串行通信总线和并行通信总线。

（2）总线的主要性能指标

总线的主要性能指标包括：总线宽度、总线带宽、总线时序等。

【例 4-15】 假设某系统总线在一个总线周期中并行传输 4 字节信息，一个总线周期占用 2 个时钟周期，总线时钟频率为 10 MHz，则总线带宽是（　　）。

A. 10 MBps　　B. 20 MBps　　C. 40 MBps　　D. 80 MBps

答案：B。

2．总线仲裁

总线主设备：获得总线控制权的设备。

总线从设备：被主设备访问的设备。

（1）集中仲裁方式

集中仲裁方式是指总线控制逻辑集中在某一控制部件中。

1）链式查询

① 基本原理：每个连接在总线上的设备有相应的逻辑，能够产生请求信号，在获准使

用总线的情况下能够接收允许信号；允许信号由总线控制逻辑发出，在各部件之间串行传送，如图 4.10 所示。

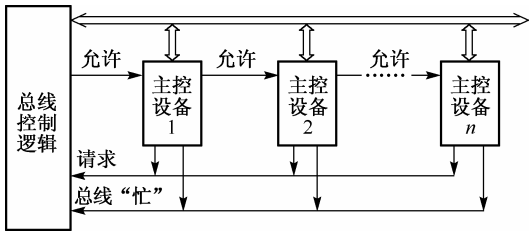


图 4.10　链式查询方式

② 链式仲裁机制：先请求者优先，并占有总线；多个设备同时请求时，逻辑上越靠近总线控制逻辑的设备，优先级越高，并占有总线。

③ 优缺点：线路简单，速度慢，对电路故障敏感，优先级固定。

2）计数器查询

① 基本原理：为每个主设备分配一个地址号，每个主设备接口中具有一个地址号识别电路；系统控制逻辑中设置一计数器，在收到请求后，该计数器开始计数，直到计数值等于提出请求的设备的设备号为止，如图4.11 所示。

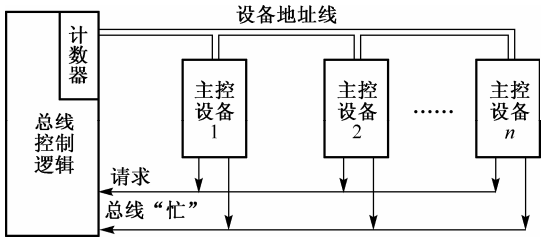


图 4.11　计数器查询方式

② 优先级的确定：

⊙ 计数器从“0”开始作加“1”计数，则设备号越小，优先级越高。

⊙ 计数器从最大值开始作减“1”计数，则设备号越大，优先级越高。

⊙ 计数器从上次中断值开始计数，则为动态优先级。

3）独立请求

① 基本原理：并行仲裁方式，“请求”信号与“允许”信号相互独立，直接送到总线控制逻辑，不需逐级传递，速度快。

② 优先级的确定

⊙ 总线控制逻辑内置优先权算法；

⊙ 系统控制逻辑内置硬件优先权排队电路。

（2）分布仲裁方式

判优硬件分布在各个总线主设备中。

① 基本原理：如图4.12 所示，为每个主设备分配一个优先权编码，每个主设备设置有一

个判优器；主设备提出请求时，将优先权编码送往判优器，同时将该编码送往一个公共的比较器，与其他请求设备的优先权编码进行比较。并产生结果 AP（收到的最大优先权编码）；提出请求的设备判优器读回比较结果 AP，并与自身优先权编码 APi 比较：

如果 APi = AP，则该判优器产生“允许”信号，对应设备占有总线；

如果 APi < AP，则该判优器不产生“允许”信号，对应设备不能占有总线。

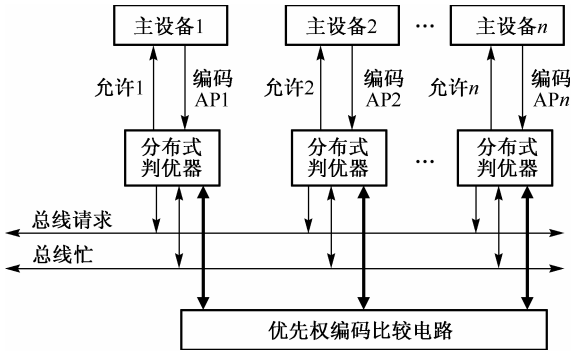


图 4.12　分布仲裁方式

3．总线操作

（1）通信方式

同步通信：通信双方由统一时标控制数据传送。适于总线长度较短，挂接部件速度相近的系统。

异步通信：无统一时标，通信联络控制信号采用应答方式，又分为不互锁、半互锁、全互锁（可靠性最高）。

半同步通信：通信双方由统一时标控制数据传送，允许根据需要改变传送时间。

（2）信息传送方式

串行传送：信息逐位分时传送。

并行传送：信息每位同时传送。

串并行传送：信息分组（组内并行、组间串行）传送。

全双工传送：信息可以双向同时传送。

半双工传送：信息可以双向传送，但不能同时双向传送。

4.1.5　输入/输出系统

1．I/O设备

（1）键盘

编码键盘：采用硬件线路实现键盘编码。

非编码键盘：利用简单的硬件和专用软件识别按键的位置，提供位置码；再由处理器执行查表程序，将位置码转换成 ASCII 码。

（2）显示器

1）有关术语

分辨率：显示器所能表示的像素个数。

水平分辨率：一条扫描线上的像点数。

垂直分辨率：一帧画面的扫描线数。

灰度级：像素点的亮暗差别或不同颜色。当灰度级设定为 24 位时，称为真彩，可同屏显示 2^{24} 种颜色。

帧频：每秒刷新一帧画面的次数。

场频：每秒从上到下刷新的次数。逐行扫描时，场频等于帧频

行频：每秒从左到右刷新一条扫描线的次数。行频=场频×行数。

视频：点脉冲的频率。视频=分辨率×帧频。

带宽：每秒扫描的总像素数。带宽=水平分辨率×垂直分辨率×场频。

2）显示存储器与字符发生器

显示存储器：存放待显示的字符的 ASCII 码，其容量由一帧画面能显示的字符数决定。

字符发生器：存放字符的点阵信息，其容量与显示器所能显示的字符种类有关。

3）计数器及其分频设置

点计数器：记录每个字的横向点数，(7+2)∶1（7 为显示点数，2 为横向间隔点数）。

字计数器：记录每排的字数，(80+*m*)∶1（80 为字数，*m* 为屏幕左右边缘过量扫描字数）。

行计数器：记录每个字的扫描行数，(9+3)∶1（9 为行数，3 为纵向间隔行数）。

排计数器：记录每屏字符的排数，(25+*n*)∶1（25 为字符排数，*n* 为屏幕上下边缘过量扫描排数）。

（3）磁盘存储器

主要技术指标：

① 存储密度：磁盘单位面积能记录的二进制信息量，包括道密度和位密度。各道周长不同，各道容量相同，内圈位密度最大，外圈位密度最小。

② 存储容量：存储器可以存储的总字节数，包括格式化容量和非格式化容量。

③平均存取时间：从读写命令发出，到开始从盘片读出或写入信息所需的时间，等于平均寻道时间+平均等待时间。

④数据传输率：单位时间向主机传送数据的字节数，等于磁盘转速×磁道容量（Bps）。

2．I/O接口

（1）接口的功能和组成

接口功能：识别设备、数据缓冲、数据转换、传送主机命令、反映设备状态、时序控制、中断或 DMA 功能。

接口组成：地址译码器、数据缓冲器、命令字寄存器、状态字寄存器、数据格式转换线路、控制逻辑等。

（2）接口的编址方式

接口的编址即是给 I/O 端口（接口中的寄存器）分配地址。主要有两种方式：统一编址

方式，即 I/O 端口和主存单元统一分配地址；单独编址方式，即 I/O 端口与主存单元分别分配地址。

3．I/O方式

（1）程序查询方式

程序查询方式是指用现行程序实现主机与外设的信息交换。CPU 用 I/O 指令查询外设状态，等待外设工作完成，便可与之交换数据。

（2）中断方式

中断是指 CPU 暂时中止现行程序的执行，转去执行为某个随机事态服务的中断处理程序，处理完毕后自动恢复原程序的执行。

有如下两种情况出现是，产生中断请求：外设工作完成，“完成”标志为 1；或者，CPU 允许请求，“屏蔽”标志为 0。

中断响应的条件包括：外设有请求，且未被屏蔽；CPU 开中断；一条指令（非停机指令）结束；无故障、DMA 等优先级更高的请求。

中断响应的过程如图 4.13 所示。

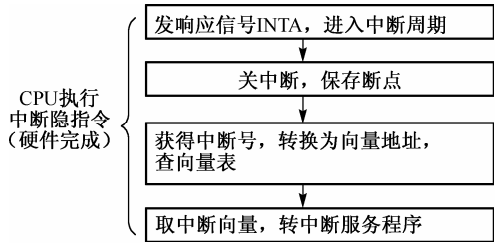


图 4.13　中断响应过程

中断处理的方式有如下两种（如图 4.14 所示）。

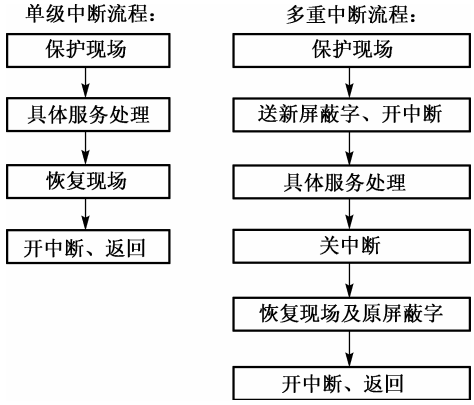


图 4.14　中断处理方式

单级中断：CPU 响应后只处理一个中断源的请求，处理完毕后才能响应新的请求。

多重中断：在某次中断服务过程中，允许响应处理更高级别的中断请求。

【例 4-16】 单级中断系统中，中断服务程序内的执行顺序是（　　）。

Ⅰ．保护现场　　Ⅱ．开中断　　Ⅲ．关中断　　Ⅳ．保存断点

Ⅴ．中断事件处理　　Ⅵ．恢复现场　　Ⅶ．中断返回

A．Ⅰ→Ⅴ→Ⅵ→Ⅱ→Ⅶ　　B．Ⅲ→Ⅰ→Ⅴ→Ⅶ

C．Ⅲ→Ⅳ→Ⅴ→Ⅵ→Ⅶ　　D．Ⅳ→Ⅰ→Ⅴ→Ⅵ→Ⅶ

答案：A。

注意题目问的是中断服务程序内的执行顺序，不是整个中断过程。

（3）DMA 方式

DMA 是指直接依靠硬件实现主存与 I/O 间的数据传送，传送期间不需 CPU 程序干预。

DMA 的传送过程如下：

① 程序准备：程序实现初始化。

② DMA 传送：DMA 控制器掌握总线权，实现 M→I/O。

③ 善后处理：中断处理程序判断传送的正误。

【例 4-17】 某计算机的 CPU 主频为 500MHz，CPI 为 5（即执行每条指令平均需 5 个时钟周期）。假定某外设的数据传输率为 0.5MBps，采用中断方式与主机进行数据传送，以 32 位为传输单位，对应的中断服务程序包含 18 条指令，中断服务的其他开销相当于 2 条指令的执行时间。请回答下列问题，要求给出计算过程。

（1）在中断方式下，CPU 用于该外设 I/O 的时间占整个 CPU 时间的百分比是多少？

（2）当该外设的数据传输率达到 5 MBps 时，改用 DMA 方式传送数据。假定每次 DMA 传送块大小为 5000B，且 DMA 预处理和后处理的总开销为 500 个时钟周期，则 CPU 用于该外设 I/O 的时间占整个 CPU 时间的百分比是多少？（假设 DMA 与 CPU 之间没有访存冲突）

解答:

（1）传送一个数据需 20 条指令，需 100 个时钟周期；每秒传送 0.5MB/4B=0.125×10^6 个数据，需 12.5×10^6 个时钟周期；CPU 时间为每秒 500×10^6 个时钟周期；所占百分比为 12.5M/500M=0.025=2.5%。

（2）每秒传送的数据块数量为 5MB/5000B=1000 个；DMA 传送的总开销为每秒 500×1000=0.5×10^6 个时钟周期；CPU 时间为每秒 500×10^6 个时钟周期；所占百分比为 0.5M/500M=0.001=0.1%。

4.2 考研真题解答

4.2.1 电子科技大学 2009 年计算机组成原理复试

1．复试题目

一、单项选择题（根据每小题题干内容，从四个备选答案中选择一个正确答案。每小题 2 分，共 20 分）：

1．在微程序控制方式中，微命令由（　　）产生。

① 程序状态字　　② 微指令　　③ 机器指令　　④ 控制程序

2．半导体存储器采用（　　）。

① 随机存取方式　　② 顺序存取方式

③ 直接存取方式　　④ 上述任意一种存取方式

3．在异步总线中，传送操作（　　）。

① 由设备控制器控制　　② 按需分配时间

③ 由统一时序信号控制　　④ 由 CPU 控制

4．同步控制方式在实际应用中所表现的主要特点是（　　）。

① 指令周期长度固定　　② 工作周期长度固定

③ 总线周期长度固定　　④ 时钟周期长度固定

5．主设备是指（　　）。

① 发送信息的设备　　② 接收信息的设备

③ 掌握总线权的设备　　④ 主要的设备

6．动态存储器依靠（　　）存储信息。

① 双稳态电路　② 门电路　③ 电源电流　④ 电容电荷

7．按数据传送格式划分，常将总线分为（　　）。

① 并行总线与串行总线　　② 同步总线与异步总线

③ 系统总线与外总线　　④ 存储总线与 I/O 总线

8．当 CPU 处于开中断状态时，（　　）。

① 允许外设提中断请求　　② 不允许外设提中断请求

③ 允许响应中断请求　　④ 不允许响应中断请求

9．在 DMA 方式的数据传送阶段，总线控制权由（　　）掌握。

① CPU　② DMA 控制器　③ 总线控制器　④ 外部设备

10．CPU 可以按地址随机访问的存储器是（　　）。

① 主存　② 光盘　③ 磁盘　④ 磁带

二、判断题（下列说法有的正确，有的错误，请作出正/误判断。每小题 2 分，共 20 分）：

1．在浮点加减运算中，对阶时既可以将小阶增大，也可以将大阶减小。

2．中断向量表中存放的是中断服务程序。

3．串行接口与系统总线之间一般采用串行方式传送数据。

4．采用隐地址可以减少指令中的地址个数。

5．CPU 通常在一个总线周期结束时响应中断请求。

6．在微程序控制方式中，一条机器指令由若干条微指令解释执行。

7．补码除法与原码除法相反，够减商 0，不够减商 1。

8．程序状态字的内容可以包括外部设备工作状态。

9．采用组合逻辑控制方式便于修改与扩展指令系统的功能。

10．DMA 方式具有随机性。

三、简答题（每小题 6 分，共 60 分）：

1．在浮点运算中，什么情况下需要进行左移规格化？如何操作？

2．DMA 方式包含哪三个阶段？各阶段分别完成哪些操作？

3．动态存储器为什么要进行刷新？采用哪种刷新方式可以既不影响 CPU 访存，又不影响存取周期？如何安排？

4．在中断接口中通常设置有寄存器选择逻辑和中断控制器，它们各具有哪些功能？

5．试比较组合逻辑控制方式和微程序控制方式的主要优、缺点。

6．什么是总线？系统总线上传送的信息通常分为哪三类？

7．试举两例说明同步控制方式在实际应用中的变化。

8．在单级中断方式下，中断服务程序应完成哪些工作？

9．在写磁盘和读磁盘过程中，磁盘适配器分别在扇区缓冲器处于什么状态（满/空）时向主机发出 DMA 请求？

10．什么是总线数据传输率？已知 PCI 总线的时钟频率为 33 MHz，总线位数为 32 位。PCI 总线的数据传输率为多少 MBps？

四、CPU 部分（40 分）：

模型机数据通路结构由 CPU 内总线、计算逻运算部件 ALU、输入选择器 A 和 B、输出移位器、通用寄存器 R0~R3、暂存器 C 和 D、地址寄存器 MAR、数据缓冲寄存器 MDR、指令寄存器 IR、程序计数器 PC、堆栈指针 SP 和程序状态寄存器 PSW 组成。

1．分别说明 IR、PC、SP 和 PSW 等寄存器的作用。

2．用寄存器传送语句（如 PC→MAR）拟出下述指令从取指到执行的流程，并回答问题。

（1）传送指令 MOV　(R0), (SP)+;

其源地址采用自增型寄存器间址，目的地址采用寄存器间址。

若取指前 PC 的内容为 5，则该指令执行后 PC 的内容为多少？

（2）加法指令 ADD　–(R1), X(R2);

其源地址采用变址寻址（形式地址存放在现行指令所在单元的下一个单元中），目的地址采用自减型寄存器间址。

若取指前 PC 的内容为 5，则该指令执行后 PC 的内容为多少？

（3）转子指令 JSR　R3;

子程序入口地址采用寄存器寻址。

若取指前 PC 的内容为 5，R3 的内容为 20，则该指令执行后 PC 的内容为多少？

五、存储器部分（25 分）：

某存储器容量为 9 KB，其中 ROM 区 4 KB，用 EPROM 芯片（4K×2 位/片）组成；RAM 区 5 KB，用 SRAM 芯片（4K×8 位/片和 1K×8 位/片）组成。地址总线 A13～A0（A0 为最低位）。

据存储器容量，EPROM 芯片和 SRAM 芯片各需多少片？分别选择一个正确答案。

（1）EPROM 芯片（　　）。

（2）SRAM 芯片（　　）。

① 2 片　　② 4 片　　③ 8 片　　④ 12 片

2．EPROM 芯片和 SRAM 芯片各连入哪几根地址线？分别选择一个正确答案。

（1）EPROM 芯片（　　）。

（2）4KSRAM 芯片（　　）。

（3）1KSRAM 芯片（　　）。

① A10～A0　　② A12～A0　　③ A11～A0　　④ A9～A0

3．需设置（　　）个片选信号？

① 1　　② 2　　③ 3　　④ 4

4．分别写出这些片选信号的逻辑式。

六、显示器部分（35 分）：

某 CRT 显示器按字符方式工作，每帧显示 30 行×50 列字符，字符点阵 8（横）×10（纵），横向间隔 2 点，纵向间隔 5 线。该显示器可显示 64 种字符。

1. 显示缓冲存储器 VRAM 的内容是什么？其基本容量为多少字节（不考虑字符属性）？

2．字符发生器 ROM 的内容是什么？其容量为多少字节？

3．需设置几个同步计数器？各计数器分频关系如何安排（不考虑回扫及屏幕边缘过量扫描所需的时间）？

4．何时访问 VRAM？何时发一次水平同步信号？何时发一次垂直同步信号？

5．若帧频为 60 Hz，则显示器点频应为多少 MHz？

2．参考答案

一、单项选择题（根据每小题题干内容，从四个备选答案中选择一个正确答案。每小题 2 分，共 20 分）

1．在微程序控制方式中，微命令由（　②　）产生。

① 程序状态字　　② 微指令　　③ 机器指令　　④ 控制程序

2．半导体存储器采用（　①　）。

① 随机存取方式　　② 顺序存取方式

③ 直接存取方式　　④ 上述任意一种存取方式

3．在异步总线中，传送操作（　②　）。

① 由设备控制器控制　　② 按需分配时间

③ 由统一时序信号控制　　④ 由 CPU 控制

4．同步控制方式在实际应用中所表现的主要特点是（　④　）。

① 指令周期长度固定　　② 工作周期长度固定

③ 总线周期长度固定　　④ 时钟周期长度固定

5．主设备是指（　③　）。

① 发送信息的设备　　② 接收信息的设备

③ 掌握总线权的设备　　④ 主要的设备

6．动态存储器依靠（　④　）存储信息。

① 双稳态电路　　② 门电路　　③ 电源电流　　④ 电容电荷

7．按数据传送格式划分，常将总线分为（　①　）。

① 并行总线与串行总线　　② 同步总线与异步总线

③ 系统总线与外总线　　④ 存储总线与 I/O 总线

8．当 CPU 处于开中断状态时，（　③　）。

① 允许外设提中断请求　　② 不允许外设提中断请求

③ 允许响应中断请求　　④ 不允许响应中断请求

9．在 DMA 方式的数据传送阶段，总线控制权由（　②　）掌握。

① CPU　② DMA 控制器　③ 总线控制器　④ 外部设备

10．CPU 可以按地址随机访问的存储器是（　①　）。

① 主存　② 光盘　③ 磁盘　④ 磁带

二、判断题（下列说法有的正确，有的错误，请作出正/误判断。每小题 2 分，共 20 分）：

1．在浮点加减运算中，对阶时既可以将小阶增大，也可以将大阶减小。　**错**

2．中断向量表中存放的是中断服务程序。　**错**

3．串行接口与系统总线之间一般采用串行方式传送数据。　**错**

4．采用隐地址可以减少指令中的地址个数。　**正**

5．CPU 通常在一个总线周期结束时响应中断请求。　**错**

6．在微程序控制方式中，一条机器指令由若干条微指令解释执行。　**正**

7．补码除法与原码除法相反，够减商 0，不够减商 1。　**错**

8．程序状态字的内容可以包括外部设备工作状态。　**错**

9．采用组合逻辑控制方式便于修改与扩展指令系统的功能。　**错**

10．DMA 方式具有随机性。　**正**

三、简答题（每小题 6 分，共 60 分）：

1．在浮点运算中，什么情况下需要进行左移规格化？如何操作？

【答】

当尾数绝对值小于 1/2 时需要进行左移规格化，尾数左移，阶码减小。

2．DMA 方式包含哪三个阶段？各阶段分别完成哪些操作？

【答】

初始化阶段：CPU 传送初始化信息。

数据传送阶段：DMA 控制器掌握总线权，控制传送。

结束阶段：CPU 执行中断程序，进行善后处理。

3．动态存储器为什么要进行刷新？采用哪种刷新方式可以既不影响 CPU 访存，又不影响存取周期？如何安排？

【答】

动态存储器靠电容电荷存储信息，时间一长，电荷可能泄放，需定期补充电荷（刷新）。

异步刷新：将各刷新周期分散安排在 2ms 间隔内。

4．在中断接口中通常设置有寄存器选择逻辑和中断控制器，它们各具有哪些功能？

【答】

前者选择 I/O 端口供 CPU 访问。后者汇集中断请求，屏蔽．判优，送出公共请求；CPU 响应后，送出中断号。

5．试比较组合逻辑控制方式和微程序控制方式的主要优、缺点。

【答】

前者优点：产生微命令的速度快。缺点：设计不规整，难于修改．扩展指令功能。

后者优点：设计规整，易于修改．扩展指令功能。缺点：速度较慢。

6．什么是总线？系统总线上传送的信息通常分为哪三类？

【答】

一组能为多个部件分时共享的信息传送线路。地址．数据．控制信息。

7．试举两例说明同步控制方式在实际应用中的变化。

【答】

不同指令可以安排不同的时钟周期数；总线周期中可以插入延长周期。

8．在单级中断方式下，中断服务程序应完成哪些工作？

【答】

保护现场，中断处理，恢复现场，开中断，返回。

9．在写磁盘和读磁盘过程中，磁盘适配器分别在扇区缓冲器处于什么状态（满/空）时向主机发出 DMA 请求？

【答】

写磁盘时扇区缓冲器空，或读磁盘时扇区缓冲器满，提出 DMA 请求。

10．什么是总线数据传输率？已知 PCI 总线的时钟频率为 33 MHz，总线位数为 32 位。PCI 总线的数据传输率为多少 MBps？

【答】

总线每秒传送的字节数。PCI 带宽=33×32÷8=132 MB/s。

四、CPU 部分（40 分）：

1．分别说明 IR、PC、SP 和 PSW 等寄存器的作用。　　10 分

【答】

IR：存放现行指令。

PC：存放指令的地址。

SP：指示栈顶的位置。

PSW：反映程序的运行状态。

2．用寄存器传送语句（如 PC→MAR）拟出下述指令从取指到执行的流程，并回答问题。

【答】

（1）传送指令 MOV　(R0), (SP)+;

其源地址采用自增型寄存器间址，目的地址采用寄存器间址。

若取指前 PC 的内容为 5，则该指令执行后 PC 的内容为多少？

FT：M→IR，PC+1→PC

ST：SP→MAR，M→MDR→C，SP+1→SP

DT：R0→MAR

ET：C→MDR，MDR→M，PC→MAR　　8 分

PC 的内容为 6。　　2 分

（2）加法指令 ADD　–(R1), X(R2);

其源地址采用变址寻址（形式地址存放在现行指令所在单元的下一个单元中），目的地址采用自减型寄存器间址。

若取指前 PC 的内容为 5，则该指令执行后 PC 的内容为多少？

FT：M→IR，PC+1→PC

ST：PC→MAR，M→MDR→C，PC+1→PC，C+R2→MAR，M→MDR→C

DT：R1-1→R1，MAR，M→MDR→D

ET：C+D→MDR，MDR→M，PC→MAR　　10 分

PC 的内容为 7。　　2 分

（3）转子指令 JSR　R3;

子程序入口地址采用寄存器寻址。

若取指前 PC 的内容为 5，R3 的内容为 20，则该指令执行后 PC 的内容为多少？

FT：M→IR，PC+1→PC

ET：SP-1→SP，MAR，PC→MDR，MDR→M，R3→PC，MAR　　6 分

PC 的内容为 20。　　2 分

五、存储器部分（25 分）：

【答】

据存储器容量，EPROM 芯片和 SRAM 芯片各需多少片？分别选择一个正确答案。

（1）EPROM 芯片（　②　）

（2）SRAM 芯片（　①　）　　5 分

① 2 片　　② 4 片　　③ 8 片　　④ 12 片

2．EPROM 芯片和 SRAM 芯片各连入哪几根地址线？分别选择一个正确答案。

（1）EPROM 芯片（　③　）

（2）4KSRAM 芯片（　③　）

（3）1KSRAM 芯片（　④　）　　7 分

① A10～A0　　② A12～A0　　③ A11～A0　　④ A9～A0

3．需设置（　③　）个片选信号？　　3 分

① 1　　② 2　　③ 3　　④ 4

4．分别写出这些片选信号的逻辑式。

$CS_0 = \overline{A}_{13}\overline{A}_{12}$，　$CS_1 = \overline{A}_{13}A_{12}$，　$CS_2 = A_{13}\overline{A}_{12}\overline{A}_{11}\overline{A}_{10}$　　10 分

六、显示器部分（35 分）：

【答】

1．显示缓冲存储器 VRAM 的内容是什么？其基本容量为多少字节（不考虑字符属性）？

字符编码，30×50=1500 字节。　　5 分

2．字符发生器 ROM 的内容是什么？其容量为多少字节？

字符点阵代码，10×64=640 字节。 5 分

3．需设置几个同步计数器？各计数器分频关系如何安排（不考虑回扫及屏幕边缘过量扫描所需的时间）？

4 个。点计数：10∶1；字符计数：50∶1；线计数：15∶1；行计数：30∶1 10 分

4．何时访问 VRAM？何时发一次水平同步信号？何时发一次垂直同步信号？

点计数一个循环访问 VRAM，字符计数一个循环发水平同步，行计数一个循环发垂直同步。 10

5．若帧频为 60 Hz，则显示器点频应为多少 MHz？

60×30×15×50×10=13500000 Hz=13.5 MHz 5 分

4.2.2 电子科技大学 2003 年计算机组成原理考试

1．考试题目

电子科技大学

2003 年攻读硕士学位研究生入学试题

考试科目：计算机原理（427）

注：应届考生作一(1～15)、二(1～8)、三、四、五、六题。

往届考生从一中选作 15 道题，从二中选作 8 道题，并作三、四、五、六题。

一、在每题的四个备选答案中选出一个正确答案，将其号码填入题干的括号内。（30 分）

1．在浮点加减运算中，对阶的原则是（ ）。

① 被加数向加数对齐 ② 加数向被加数对齐

③ 小阶向大阶对齐 ④ 大阶向小阶对齐

2．在补码不恢复余数除法中，（ ）。

① 余数为正商 1，为负商 0 ② 余数为正商 0，为负商 1

③ 余数与除数同号商 0，异号商 1 ④ 余数与除数同号商 1，异号商 0

3．采用隐式 I/O 指令是指（ ）。

① 用隐地址实现 I/O 操作 ② 用传送指令实现 I/O 操作

③ 用隐指令实现 I/O 操作 ④ 用通道指令实现 I/O 操作

4．在下列存储器中，允许随机访问的存储器是（ ）。

① 光盘 ② 堆栈 ③ 磁盘 ④ ROM

5．在异步总线中，传送操作（ ）。

① 由设备控制器控制 ② 按需分配时间

③ 由统一时序信号控制 ④ 由 CPU 控制

6．为了减少指令中的地址个数，应尽量采用（ ）。

① 隐地址 ② 寄存器间址 ③ 变址 ④ 寄存器寻址

7．微程序存放在（ ）。

① 磁盘中 ② 主存中 ③ 堆栈中 ④ 控存中

8．主设备是指（　　）。
① 发送信息的设备　　② 接收信息的设备
③ 掌握总线权的设备　　④ 主要的设备
9．CPU 响应 DMA 请求是在（　　）。
① 一个时钟周期结束时　　② 一个总线周期结束时
③ 一条指令结束时　　④ 一段程序结束时
10．静态 RAM 利用（　　）。
① 门电路存储信息　　② 读电流存储信息
③ 电容存储信息　　④ 触发器存储信息
11．向量表中存放（　　）。
① 向量地址　② 中断类型号　③ 服务程序入口地址　④ 控制/状态字
12．外设与主存统一编址是指（　　）。
① 每个接口占一个地址码　　② 接口中的有关寄存器各占一个地址码
③ 每台外设占一个地址码　　④ 每台外设由一个主存单元管理
13．当 CRT 显示器的点计数器计数一个循环后，（　　）。
① 发 ROM 行选信号　　② 发水平同步信号
③ 发垂直同步信号　　④ 访问显示缓冲存储器
14．在写磁盘过程中，适配器向主机发出 DMA 请求是在（　　）。
① 启动磁盘时　　② 寻道完成时
③ 扇区缓冲器满时　　④ 扇区缓冲器空时
15．扩展同步总线（　　）。
① 无时钟周期划分　　② 无总线周期划分
③ 允许时钟周期长度可变　　④ 允许总线周期长度可变
16．在微程序控制方式中，通常一条微指令控制执行（　　）。
① 一个微操作　② 一步操作　③ 一条机器指令　④ 一段工作程序
17．当采用双符号位时，发生正溢的特征是双符号位为（　　）。
① 00　② 01　③ 10　④ 11
18．在键盘接口中设置移位寄存器是为了实现（　　）。
① 串→并转换　② 并→串转换　③ 字节→字转换　④ 字→字节转换
19．在三级时序系统中，电平型微命令一般维持（　　）。
① 一个脉冲宽度　　② 一个时钟周期
③ 一个工作周期　　④ 一个指令周期
20．DMA 传送通常在（　　）。
① I/O 与寄存器之间进行　　② I/O 与 CPU 之间进行
③ I/O 与主存之间进行　　④ I/O 与主机之间进行

二、简答题（40 分）：

1．组合逻辑控制器是如何产生微命令的？微程序控制器又是如何产生微命令的？

2．什么是中断方式？它与转子有何区别？

3．在浮点加减运算中，什么情况下需要左移规格化？什么情况下需要右移规格化？

4．直接存取存储器和顺序存取存储器相比较，哪一个的工作速度较快？为什么？

5．什么是串行接口？举例说明在哪些场合下需使用串行接口。

6．动态存储器为什么要定时刷新？集中．分散．异步三种刷新方式如何安排刷新周期？

7．中断屏蔽技术常用在哪些场合？试举两例说明。

8．同步控制方式有哪些主要特征？一般应用在哪些场合？

9．在调用磁盘时，一般应给出哪些寻址信息？

10．当主机连接多种外设时，如何用通用 I/O 指令实现对这些设备的具体控制？

11．简述一种软件扫描方法，使其能查找按键位置，并转换为按键编码。

12．系统总线和 CPU 内总线有什么区别？

三、CPU 部分（20 分）

模型机数据通路结构由以下部分组成：CPU 内总线、算逻运算部件 ALU、输入选择器 A 和 B、输出移位器、通用寄存器 R0~R3、暂存器 C 和 D、地址寄存器 MAR、数据缓冲寄存器 MBR、指令寄存器 IR、程序计数器 PC、堆栈指针 SP。

1．该数据通路结构框图。

2．出下述指令从读取到执行的完整流程，用寄存器传送语句（如 R0→MAR）描述。

（1）传送指令 MOV　(R0), -(SP);

源和目的分别采用寄存器间址和堆栈寻址。

（2）加法指令 ADD　(R1)+, R2;

源和目的分别采用自增型寄存器间址和寄存器寻址。

（3）转移指令 JMP　(R3);

转移地址采用寄存器间址。

四、（20 分）

已知地址总线 A15～A0（低），双向数据总线 D7～D0（低），读/写信号线 R/W。用 ROM 芯片（4 K×4 位/片）和 RAM 芯片（2 K×8 位/片）组成一个半导体存储器，按字节编址。其中 ROM 区地址从 0000H～0FFFH，RAM 区地址从 1000H～27FFH。

1．组成该存储器需用多少块 ROM 芯片和多少块 RAM 芯片？

2．各芯片需连入哪几根地址线？

3．分别写出各片选信号的逻辑式。

4．画出该存储器逻辑粗框图。图中应包括所选用的芯片、片选逻辑电路（片选低电平有效）以及各信号线的连接。

五、（20 分）

某机需扩展 4 台 I/O 设备，每台设备都可用中断方式与主机交换数据。4 台设备共用一个中断号，通过中断控制器的 IRQ2 请求输入端进行扩展。

1．为 4 台设备设计一个中断接口，画出寄存器级粗框图。

2．描述从扩展设备提出中断请求，到转入扩展设备服务程序的过程。

3．若允许多重中断，则扩展设备的服务程序应如何组织？

六．（20 分）

某 CRT 显示器每帧显示 25 行×40 列字符，每个字符由横向 5 点、纵向 7 点组成，字符横向间隔 3 点，纵向间隔 5 点。

1．显示缓冲存储器的内容是什么？基本容量有多少字节（不考虑字符属性）？

2．需设置哪几级分频计数器？按顺序画出这些计数器的连接。

3．计算出各级计数器的分频比例（假设左右边缘过量扫描和水平回扫共占 *m* 字符时间，上下边缘过量扫描和垂直回扫共占 *n* 行字符时间）。

2．参考答案

一、单选题（每小题 2 分，共 30 分）

1．（3）　2．（4）　3．（2）　4．（4）　5．（2）　6．（1）　7．（4）　8．（3）

9．（2）　10．（4）　11．（3）　12．（2）　13．（4）　14．（4）　15．（4）　16．（2）

17．（2）　18．（1）　19．（2）　20．（3）

二、简答题（每小题 5 分，共 40 分）

1．组合逻辑控制器通过组合逻辑电路产生微命令，微程序控制器通过微指令产生微命令。

2．中断方式指：CPU 暂停现行程序的执行，转去执行为某个随机事件服务的中断处理程序。处理完毕后自动恢复原来程序的执行。

中断方式具有随机性，转子无随机性。

3．当尾数绝对值小于 1/2 时需要左移规格化，尾数绝对值大于 1/2 时需要右移规格化。

4．直接存取存储器的工作速度较快。因为直接存取存储器在访问时，由读写部件直接指向存储器的一个小区域，再在该区域内顺序查找。其查找范围比顺序存取存储器小得多，所以速度较快。

5．串行接口指：接口与主机并行传送，接口与外设串行传送。当设备本身按串行方式工作（如磁盘），或距离主机较远（如通信设备），或需减少传送线（如键盘）时，需使用串行接口。

6．动态存储器靠电容电荷存储信息，时间一长，电荷可能泄漏，所以要定时刷新。

集中方式将所有刷新周期集中安排在 2 ms 内；分散方式将所有刷新周期分散安排在各存取周期中；异步方式将所有刷新周期分散安排在 2 ms 内。

7．例 1：屏蔽优先级较高的请求，以动态改变外部请求的优先级。

例 2：屏蔽同级或较低级的请求，以实现多重中断。

8．主要特征：有明显时序时间划分，时钟周期时间固定，各操作的衔接和各部件的数据传送受严格同步定时控制。用于 CPU 内部、设备内部、系统总线操作。

9．驱动器号、磁头号、磁道号、起始扇区号、扇区数。

10．接口中设置控制/状态寄存器，接收 CPU 发来的设备控制命令，或向 CPU 传送设备状态信息，以便对设备进行控制。

11．逐行扫描方法：先向行线送全 0，判列线中有无 0。如有，则说明有键按下。再逐行送 0，判哪列为 0，以确定按键位置。按位置码查表，获得相应编码。

12．系统总线连接系统中各功能部件，分为地址、数据、控制三类信号线，同步控制或异步控制；内总线连接 CPU 内部各寄存器与算逻部件，为数据线，同步控制。

三、（20 分）

1．如图 4.15 所示。（5 分）

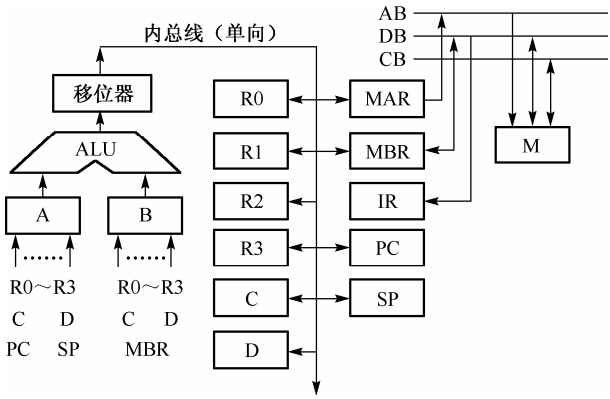


图 4.15　数据通路结构框图

2．（1）传送指令 MOV　(R0), -(SP);（7 分）

M→IR，PC+1→PC

R0→MAR

M→MBR→C

SP-1→SP，MAR

C→MBR

MBR→M

PC→MAR

（2）加法指令 ADD　(R1)+, R2;（5 分）

M→IR，PC+1→PC

R1→MAR

M→MBR→C

R1+1→R1

C+R2→R2

PC→MAR

（3）转移指令 JMP　(R3);（3 分）

M→IR，PC+1→PC

R3→MAR

M→MBR→PC，MAR

四、（20 分）

1．芯片数 （5 分）

ROM：2 片，RAM：3 片。

2．芯片地址 （2 分）

4K：A11～A0，2K：A10～A0。

3．片选逻辑式 （8 分）

CS0=A15A14A13A12（0000）

CS1=A15A14A13A12A11（00010）

CS2=A15A14A13A12A11（00011）

CS3=A15A14A13A12A11（00100）

4．框图如图 4.16 所示。 （5 分）

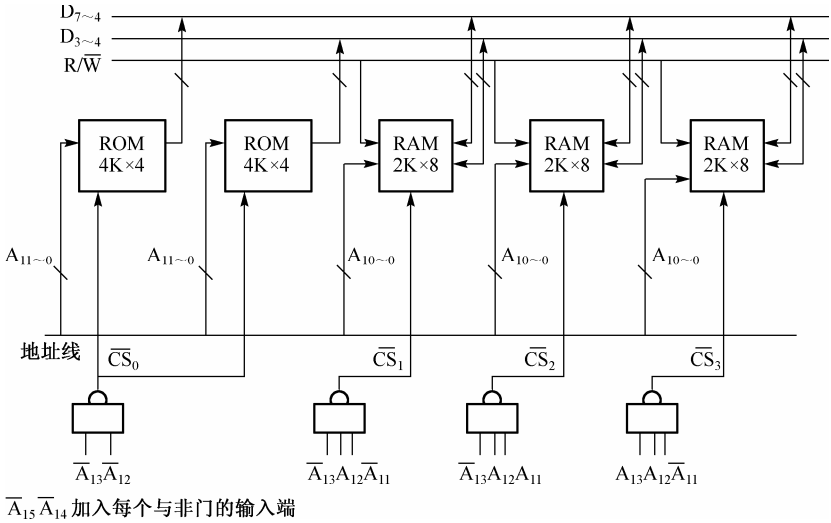


图 4.16　存储器逻辑粗框图

五、（20 分）

1．如图 4.17 所示。 （6 分）

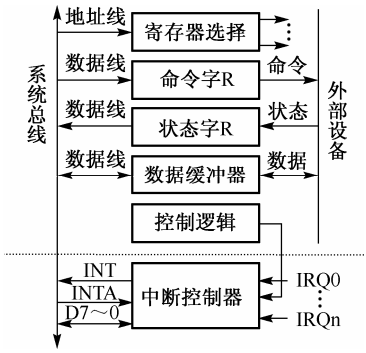


图 4.17　寄存器级粗框图

2．（8 分）

扩展设备提出中断请求，送入 IRQ2，经屏蔽．判优，由中断控制器向 CPU 送出公共请求。CPU 响应，从中断控制器取回 IRQ2 中断号（假设 IRQ2 优先级最高），转化为向量地址后访问向量表。从表中取得 IRQ2 服务程序入口，转入 IRQ2 服务程序。在服务程序中查询扩展设备的状态，再转入扩展设备的服务程序。

3．（6 分）

① 保护现场。
② 送新屏蔽字，开中断。
③ 具体的中断处理。
④ 关中断，恢复现场。
⑤ 开中断，返回。

六、（20 分）

1．（6 分）

内容是字符编码，基本容量有 1000 字节。

2．（6 分）

需设置 4 级分频计数器：点计数器→字符计数器→线计数器→行计数器。

3．（8 分）

点计数器：8∶1　　字符计数器：$(40+m)$∶1

线计数器：12∶1　　行计数器：$(25+n)$∶1